Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____
(повна назва)

Кафедра _____Програмної інженерії_____
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти – другий (магістерський)

Методи кластеризації даних Е-магазинів з підтримки економічно вигідних рішень для клієнтів за модою та найкращою ціною
(тема)

Виконала: студент 2 курсу, групи ІПЗм-18-4___
_____Яременко М.М._____
(прізвище, ініціали)
спеціальності 121- Інженерія програмного забезпечення____
(код і повна назва спеціальності)
____Освітньо-наукової програми_____
(тип програми)
____Інженерія програмного забезпечення_____
(повна назва освітньої програми)

Керівник _____доц. Ревенчук І.А._____
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри, проф. _____ З.В.Дудар

2020 р.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет	Комп'ютерних наук
Кафедра	Програмної інженерії
Рівень вищої освіти	другий (магістерський)
Спеціальність	121 - Інженерія програмного забезпечення
Освітньо-наукова програма	Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)
«_____»_____ 20 ___ р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові _Яременко Мирославі Максимівні_____
(прізвище, ім'я, по батькові)

1. Тема роботи _Методи кластеризації даних Е-магазинів з підтримки економічно вигідних рішень для клієнтів за модою та найкращою ціною.

затверджена наказом по університету від 27.03.2020 р. № 473

2. Термін подання студентом роботи до екзаменаційної комісії 5 червня 2020 р.

3. Вихідні дані до роботи електронні ресурси за обраною тематикою, мінімальні вимоги до функціональності програми, загальні вимоги до архітектури програмного забезпечення, алгоритми аналізу даних, середовище проектування WebStorm.

4. Перелік питань, що потрібно опрацювати в роботі аналіз предметної галузі, мета роботи, дослідження методів пошуку зображень за методами кластеризації, алгоритми кластеризації, моделювання програмного забезпечення.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) *скріншоти систем аналогів, діаграма напрямків машинного навчання, діаграми алгоритмів, блок-схема алгоритму кластеризації, скріншоти інтерфейсу, скріншоти вхідних даних, скріншоти коду, слайди презентації, апробація результатів роботи*.

6 Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спецчастина	доц. Ревенчук І.А.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка *
1.	Аналіз предметної галузі	27 січня 2020 р.	
2.	Огляд існуючих методів	10 лютого 2020 р.	
3.	Огляд існуючих алгоритмів кластеризації	09 березня 2020 р.	
4.	Підготовка пояснювальної записки	23 березня 2020 р.	
5.	Спецчастина	20 травня 2020 р.	
6.	Підготовка презентації та доповіді	1 червня 2020 р.	
7.	Попередній захист	5 червня 2020р.	
8.	Нормоконтроль, рецензування	6 червня 2020 р.	
9.	Занесення диплома в електронний архів	10 червня 2020 р.	
10.	Допуск до захисту у зав. кафедри	11 червня 2020 р.	
* заповнюється вручну після виконання чергового пункту			

Дата видачі завдання 30.03.2020 р.

Студент _____
(підпис)

Керівник роботи _____ __доц. Ревенчук І.А.__
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Атестаційна робота магістра містить: 78 с., 21 рис., 6 форм., 23 джерел.

АНАЛІЗ ДАНИХ, КЛАСТЕРИЗАЦІЯ, ЗНАХОДЖЕННЯ ЗОБРАЖЕНЬ, СЕГМЕНТАЦІЯ ЗОБРАЖЕНЬ, МОДА.

Об’єкт – дослідження методів кластеризації даних для Е-магазинів з підтримки економічно вигідних рішень для клієнтів за модою та найкращою ціною.

Мета роботи – дослідити методи кластеризації даних для формування економічно вигідних рішень для Е-магазинів за модою та найкращою ціною.

В роботі проведений аналіз предметної галузі, досліджені алгоритми кластеризації даних, сегментації зображень, концепція економічних досліджень та обраний метод кластеризації даних для вирішення задачі.

Master final work contains: 78p., 21 pic., 6 formulas, 23 sources.

DATA MINING, CLUSTERIZATION, IMAGE DETECTION, IMAGE SEGMENTATION, MODE.

The objective – investigation of e-commerce data clustering techniques for supporting cost-effective customer solutions with mode and best price criteria. The purpose – to explore data clustering methods to generate cost-effective E-commerce solutions using mode and best price criteria.

The article analyzes the subject area, investigates algorithms for data clustering, the concept of economic research, and the method of data clustering for image segmentation purpose for problem solution.

ЗМІСТ

ВСТУП

У наш час най усе більшу актуальність обирають інтернет-платформи для продажу товарів. Більшість власників малого, середнього та крупного бізнесу намагаються створити онлайн платформи для продажу своїх послуг та товарів. Все існує суттєва конкуренція у даній сфер, тому дуже важливо, щоб дана система працювала якісно, представляла клієнту найкращі варіанти, для цього данні повинні бути добре структурованими та упорядкованими.

Онлайн агрегатори – це платформи, на яких можна порівнювати пропозиції за ціною на один і той же товар від різних продавців. Вигоду, таким чином, отримують як покупці, так і продавці. Користувачі цінують агрегатори не тільки через можливість дешевше купити потрібний продукт, а й за те, що вони економлять їх час. Адже навіть якщо просто спробувати порівняти ціни на сайтах кількох найбільших інтернет-магазинів, це може зайняти велику кількість часу. За допомогою таких сайтів можна істотно збільшити кількість можливих точок контакту з потенційними клієнтами, що робить їх потужним каналом інтернет-маркетингу з хорошим потенціалом. Багато власників онлайн бізнесу до сих пір не використовують агрегатори, що суттєво знижує їх заробіток. Результати пошуку користувач може сортувати за релевантністю, ціною, обсягами продажів, рейтингу продавця, а також обмежувати пошук по конкретному місту, що актуально для жителів мегаполісів та багатьма іншими критеріями, які тільки існують у сучасних агрегаторах.

Люди називають речі різними речами, а в роздрібній торгівлі найгірше, що ви можете зробити – це пошукова система не знаходить нічого клієнтові, тому що вони набрали альтернативне слово чи синонім, який не було знайдено у базі, або якщо користувач обрав не вірні критерії для пошуку. Навчання машини пошуку всіх предметів у конкретному класі вимагає, щоб ваші дані про навчання були чітко визначені. Тому дуже важливо коректно згрупувати дані, які будуть використовуватись для формування результату пошуку, то дуже важливо брати

найкращий метод групування даних. Для коректної відповіді від пошукового агрегатору також дуже важливу роль грають вхідні дані, тож необхідно витягнути параметри для пошуку із вхідних даних. Вхідними даними можуть бути текстові дані, тобто звичайний текстовий запит на пошук текстових співпадінь з назвою, або характеристикою товару, або як зображення, тому для такого випадку потрібно додатково реалізовувати розпізнавання образу, що зображений на вхідному зображенні.

Метою роботи є дослідження варіантів кластеризації даних для Е-магазинів з можливістю підтримки економічно вигідних рішень для клієнтів за модою та найкращою ціною.

В роботі проведений аналіз предметної галузі, досліджені основні методи, моделі, алгоритми аналізу даних, для побудови коректної моделі кластеризації даних із урахуванням таких параметрів, як мода та найкраща ціна, та пошуку товару за зображенням із використанням методу сегментації зображень з використанням кластеризації.

1 **АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ**

1.1 Огляд проблеми та актуальність

В епоху розвитку Internet усе більшої популярності набувають Internet-платформи з можливістю здійснити покупки онлайн. Для того, щоб мати змогу при пошуку, потрібних для покупця об’єктів, оперувати великою кількістю даних швидше - з’являється необхідність групування існуючих об'єктів по критеріям.

У наш час інтернет-покупець більш раціонально обирає товари, прагне знайти кращий продукт з оптимальною ціною, уважно прислухається до рекомендацій і відгуків інших людей. На сьогоднішній час, споживач дуже уважний – задоволення його потреб і відповідність перевагам потенційного покупця як ніколи впливає на успішність маркетингових стратегій та розвиток бізнесу в цілому. Дослідивши переваги сучасного онлайн-користувача, маркетологи BigCommerce створили корисну інфографіку [1], що оповідає про те, які фактори найбільше впливають на рішення відвідувача інтернет-магазину в процесі здійснення покупок. На рисунку 1.1 можна побачити топ 10 факторів.

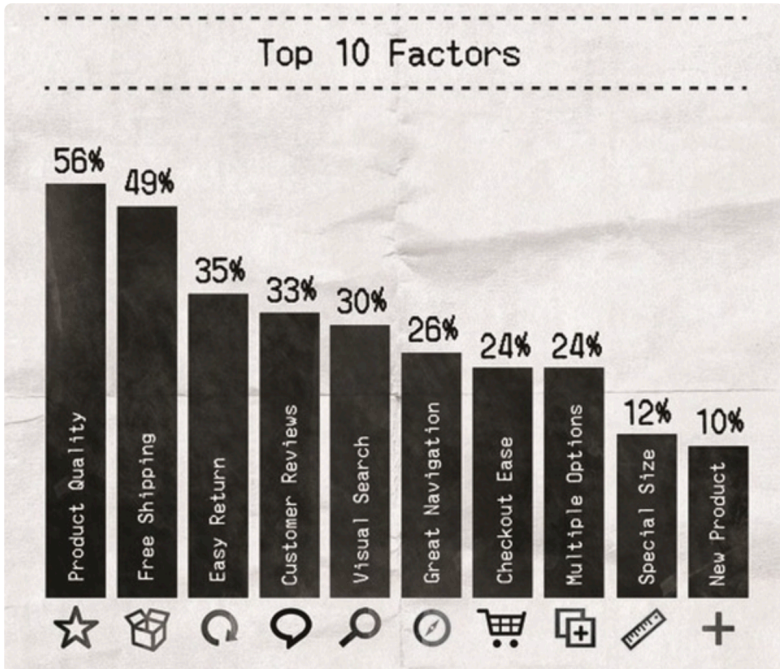

Рисунок 1.1 – Фактори впливу

Виходячи із даних, що бачимо на графіку, наступні фактори відносяться безпосередньо до юзабіліті веб-ресурсу та мають вплив на рішення користувача.

Для 56% покупців товар повинен відповідати пошуковому запиту, тобто задовольняти потреби шукача. 30% покупців надають перевагу візуально простому пошуку, у дану категорію відносять групування товарів, для легшого сприйняття. Зручну і просту навігацію інтернет-магазину цінують 26% користувачів, а 24% позитивно ставляться до надання їм кількох варіантів одного типу товару. Також для 10% споживачів важливу роль відіграє наявність нових товарів [1].

Більшість існуючих платформ пропонують величезну кількість варіантів групування та фільтрації даних. Враховуючи потреби клієнтів. Але існує багато факторів, які навпаки ускладнюють пошук у платформах із продажом товарів.

Негативний вплив на пошук за критеріями має занадто велика складність фільтрів, що використовується. Також суттєвим недоліком є те, що пошук товару за зображенням має невелика кількість платформ з продажу товарів.

На даний момент існує дуже мала кількість агрегаторів для пошуку товарів у різних інтернет магазинах за критеріями, а не за назвою чи маркою. У ході дослідницької роботи не було знайдено жодного агрегатору, який би мав змогу знайти потрібний товар за зображенням із використанням додаткових обмежуючих фільтрів, таких, як ціновий діапазон та модою, під модою у даному випадку мається на увазі найбільш популярні об’єкти.

1.2 Огляд існуючих рішень

Агрегатор – це інтернет-платформа, яка збирає всю необхідну для користувача інформацію про товар: характеристики, ціни, наявність в магазинах, відгуки і оцінки. Цей сервіс спрямований на підвищення зручності при виборі товару, а також для демонстрації всіх можливих пропозицій за конкретним запитом. При цьому варто розуміти, що самі прайс агрегатори не продають товари.

Магазини платять прайс агрегаторами за переходи або розміщення. Більше трафіку – більше залучених клієнтів.

Бажаючи заощадити, люди часто використовують прайс-агрегатори для порівняння цін в різних Інтернет-магазинах. Тому продавці, ґрунтуючись на даних агрегаторів, можуть виставляти конкурентоспроможну вартість, а також отримувати додатковий трафік цільової аудиторії, спроможної на покупку. Таким чином агрегатори залучають трафік на сайт інтернет-магазину, підвищуючи його відвідуваність, а також в якійсь мірі допомагають збільшити відсоток додаткових продажів за рахунок інтересу користувача в конкретному товару за рахунок можливості виконати порівняння цін інтернет магазинів.

«Hotline.ua» – український онлайн-сервіс для вибору товарів і порівняння цін. З 1 січня 2006 почав свою роботу інтернет-проект hotline.ua. В березні 2014 року відкрито україномовну версію сайту. Щоденна аудиторія сайту складає 100 тис. відвідувачів, щомісячна аудиторія – 1,5 млн. Вигляд сервісу представлено на рисунку 1.2.

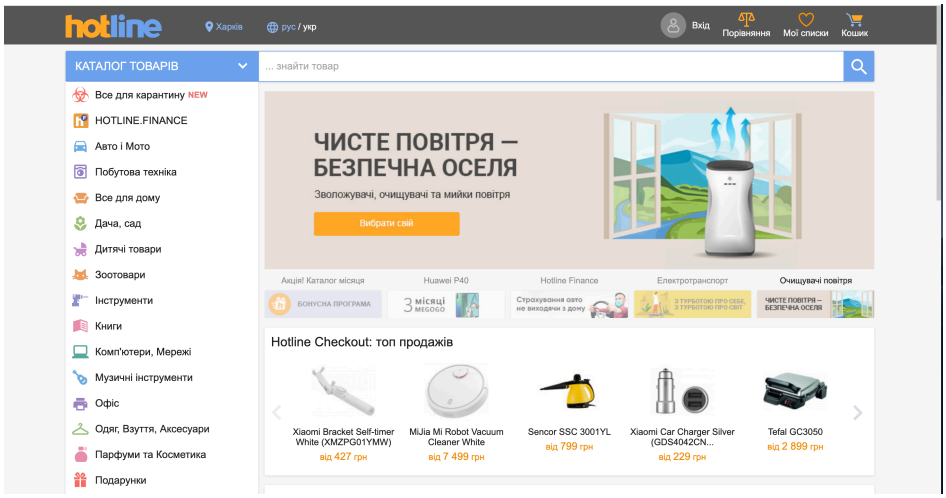

Рисунок 1.2 – «Hotline.ua»

Сервіс має велику різноманітність критеріїв для пошуку потрібного товару, але не має можливості для пошуку за заданим зображенням.

Функціонує за зразком співробітництва із магазинами, тобто власники інтернет магазинів укладають договір із сервісом для того, щоб їх товари були представлені на даному сервісі.

«Price.ua» – найбільша платформа України з просування розвитку e-commerce і всієї сфери онлайн-послуг. Існує більше 10 років на ринку, охоплення в місяць – 5-7 млн унікальних користувачів, переглядів – більше 80 млн, зареєстрованих користувачів - 1,5 млн. Входить до ТОП-у кращих онлайн-проектів України. Вигляд сервісу можна побачити на рисунку 1.3.

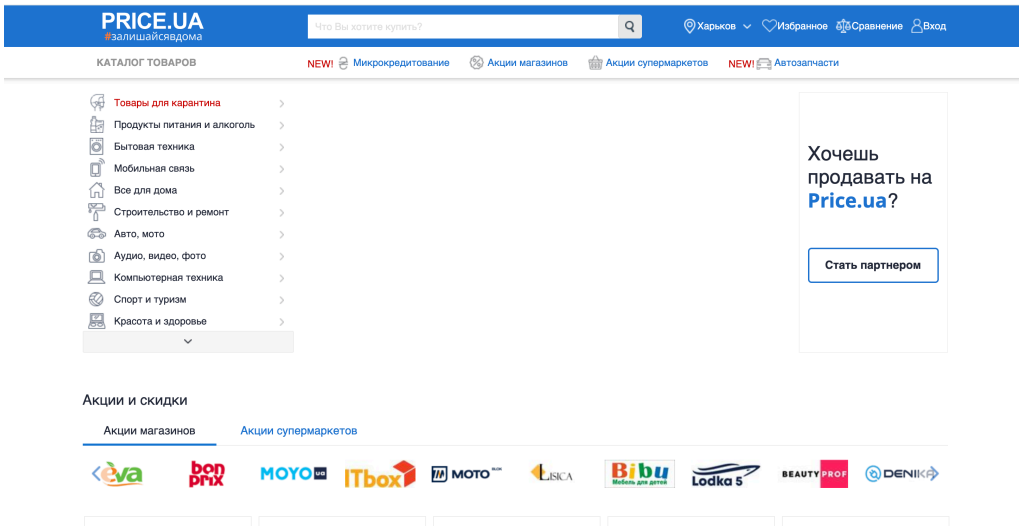

Рисунок 1.3 – «Price.ua»

Проаналізувавши даний сервіс, можна зробити висновок, що він також має велику кількість критеріїв для пошуку, але не мають змогу знаходити товар по зображенню. Також, крім технічних описів кожної моделі, можна дізнатися думку людей, які вже зробили вибір, а також залишити свій незалежний коментар в розділі «Відгуки». Для покупця «Price.ua» – це меню найвигідніших цін, для продавця – можливість зробити їх найбільш вигідними, бо вони мають змогу проаналізувати ринок та стати більш привабливими для споживачів.

«Aliexpress.com» – інтернет-магазин роздрібної торгівлі малого бізнесу Китаю. Платформа допомагає малим підприємствам продавати свою продукцію клієнтам по всьому світові і, так само як і на сайті «Amazon» або «eBay», на «AliExpress» можна знайти практично все що завгодно для продажу. Аналогічно до

«eBay», продавцями на «Aliexpress» можуть бути як компанії, так і приватні особи. «Aliexpress» напряму з'єднує китайських підприємців із споживачами. Вигляд системи можна побачити на рисунку 1.4

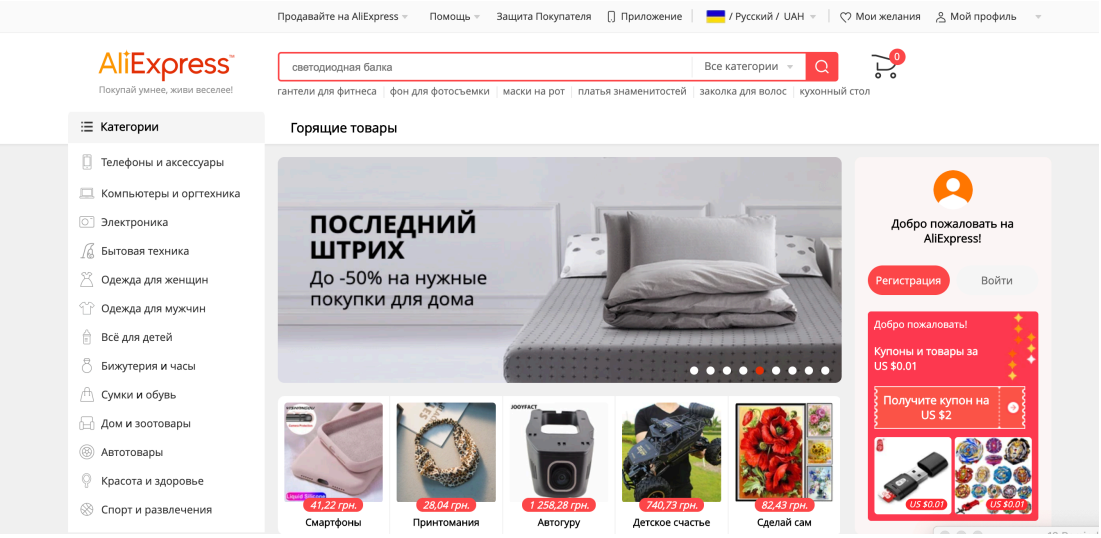

Рисунок 1.4– «Aliexpress.com»

Як і два попередніх сервіси, «Aliexpress.com» не має пошуку за зображенням у веб-сервісі, така функція існує у мобільних додатках, але вона не реалізована у вебі.

Отже, «Hotline.ua» та «Price.ua» мають функціонал пошуку товарів за критеріями, ранжування за цінами але не мають функціоналу пошуку за зображенням. У той час, як платформа «Aliexpress.com» функціонал пошуку за критеріями, ранжування за ціною, має можливість пошуку товарів за зображенням, але тільки для мобільних застосунків. Також відмінною особливістю «Aliexpress.com» є те, що він агрегує тільки підприємців які напряму співпрацюють із платформою, тобто мають свій акаунт для продажу товарів на цій платформі за зразком «Amazon» або «eBay».

Жодна із цих платформ не має функціоналу який би дозволив використовувати незалежні інтернет магазини для пошуку товарів за критеріями економічної вигідності, модою та зображенням водночас, при цьому щоб це був веб-сервіс.

1.3 Постановка задачі

Якщо розглядати проблему у цілому, то має сенс частково об'єднати функціональність вищезгаданих ресурсів і розробити такий сервіс, який мав би можливість дати користувачеві пошуковий результат за зображенням з урахуванням моди та найкращою ціною.

Для того, щоб мати змогу виконувати пошук, нам потрібні данні з різних інтренет-магазинів серед яких і буде проводитись пошук. Не всі інтернет магазини мають відкритий API, можна навіть сказати, що таких магазинів дуже мало, тому для того, щоб якимось чином нам отримувати дані з них ми повинні їх збирати власноруч, за допомогою парсерів, важливо зауважити, що для кожного магазину потрібна власна конфігурація парсеру, бо кожен сайт має власну структуру та селектори, а парсер працює за принципом обробки DOM-елементів.

Коли ми маємо дані із різних сайтів постає задача у тому, щоб їх згрупувати, але ми не можемо знати за якими критеріями буде виконуватися ця процедура, тому нам потрібно кластеризувати наші дані, які ми отримуємо після виконання парсингу. Також, для того, щоб мати змогу пошуку за зображенням, потрібно викори знайти найбільш підходящий спосіб знаходження та визначення об’єкту на зображенні та порівнянні його із іншими об’єктами при пошуку. Тобто потрібно виконати обробку зображень для даних, що були отримані при парсингу та знайти відповідний кластер для представлення результату пошуку.

Таким чином, користувач зможе отримати найбільш релевантні варіанти для вибору товарів, спираючись не тільки на критерії пошуку, та назву товару, а ще і на зовнішній вигляд, бо дуже часто буває, коли людина точно не знає назву товару и має лише зображення і їй потрібно знайти аналог, що буде дешевше, або цей же товар за найбільш вигідною ціною.

2 МЕТОДИ ПОШУКУ ЗА ЗОБРАЖЕННЯМ ІЗ ВИКОРИСАННЯМ МЕТОДІВ КЛАСТЕРИЗАЦІЇ

Люди хочуть отримувати найбільш релевантні результати, або аналогічні, схожі результати. Таким чином, задача полягає в тому, щоб дати їм вибірку з варіантів, що для них найбільш актуальною. Наприклад, скажімо, у нас є купа фотографій або продуктів, які містять різні речі, такі як взуття, сорочки, сукні, джинси, куртки і тощо. Нам потрібно якимось чином знайти на вхідному зображені об’єкт для пошуку, а потім знайти цей об’єкт серед існуючих даних.

У наш час існують поняття машинного навчання та комп’ютерного зору. Для того, щоб розібратися у цих визначеннях, спочатку потрібно звернутись до такого поняття, як штучний інтелект, бо дане поняття містить їх у собі.

Штучний інтелект – галузь у сфері комп’ютерних наук, що займається моделюванням інтелектуальної поведінки машин, передбачає здатність комп'ютерної програми або машини мислити і вчитися.

Термін «штучний інтелект» описує програму, яка намагається відтворити розумові процеси людини. Мається на увазі те, що такі дії як навчання і прийняття рішення, можуть бути виконані комп'ютерами, Отже, подібні продукти повинні мати здатність правильно інтерпретувати зовнішні дані, робити висновки опираючись на вхідні дані та використовувати набуті знання для досягнення конкретних цілей. Досконала інтелектуальна машина – інструмент, який обробляє дані із навколишнього середовища і вживає заходів, щоб максимізувати свої шанси на успіх при досягненні визначених цілей.

Із процесом вдосконалення машин, деякі процеси, що раніше, як здавалось вимагають обробки «штучного інтелекту», не включаються у визначення. Наприклад, розпізнавання символів зараз не сприймається як задача, для вирішення якої потрібен «штучний інтелект», зараз це тривіальна задача, яка вимагає тривіального вирішення [2].

Поняття «штучного інтелекту» включає в себе безліч різних областей, таких як інформатика, математика, психологія, лінгвістика та т. п. На рисунку 2.1 можна побачити діаграму, яка демонструє теми, що охоплюються поняттям штучного інтелекту.

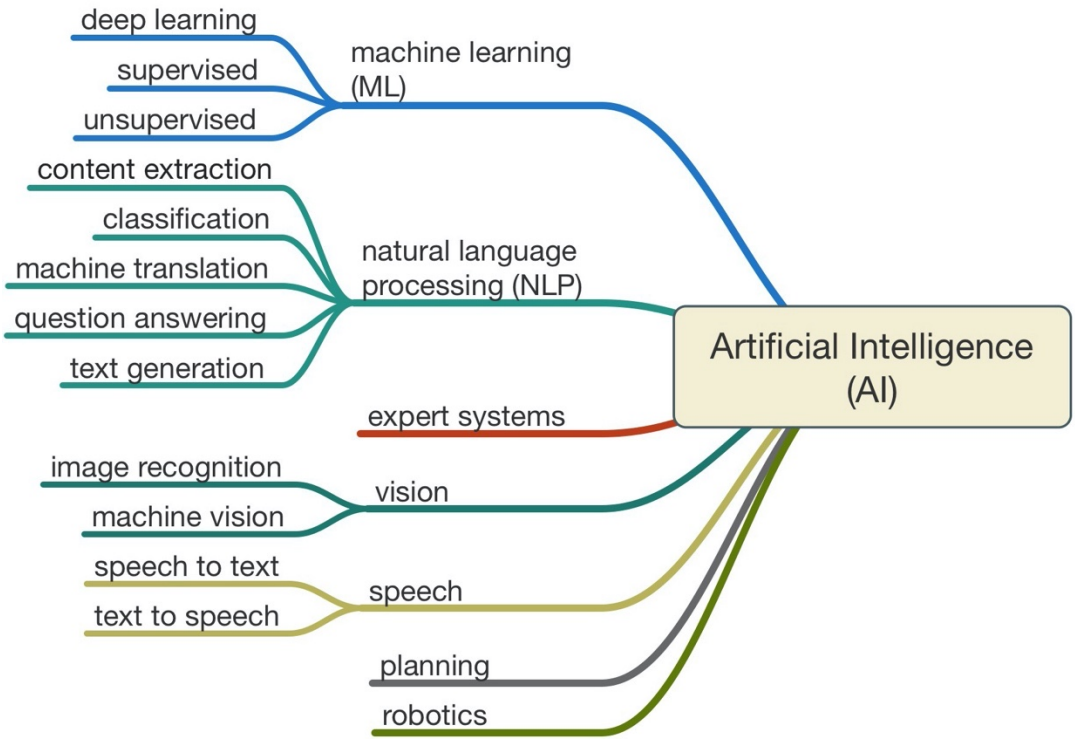

Рисунок 2.1 – Діаграма напрямків штучного інтелекту

Виходячи з діаграми, можна побачити основні напрямки у сфері штучного інтелекту:

а) машинне навчання (ML) – це метод, при якому мета та кроки, необхідні для її досягнення, вивчаються самою машиною шляхом накопичення досвіду. Наприклад, для ідентифікації простого об'єкта, такого як апельсин або помаранчевий, результат досягається не шляхом існуючого рішення, а шляхом навчання, наприклад, ми вчимо дитину, показуючи кілька різних зображень. Тобто, даємо можливість машині визначати кроки для ідентифікації об'єктів самостійно. Машинне навчання включає у себе багато підтем, які більш детально буде розглянуто нижче;

б) обробка природної мови (NLP) – Обробка природної мови є підрозділом інформатики і штучного інтелекту. NLP дозволяє комп'ютерній системі розуміти і обробляти людську мову, такий як англійська. Обробка природної мови визначається як автоматичне маніпулювання програмним забезпеченням над текстом або мово. NLP грає важливу роль в штучному інтелекті, так як без NLP машина не може виконувати людські інструкції, але за допомогою NLP ми можемо навчити систему штучного інтелекту нашої мови. Додаток обробки природної мови дозволяє користувачеві безпосередньо спілкуватися з системою своїми словами. Вхід і вихід додатків NLP можуть бути в двох форматах – це у форматі мови або тексту. Одним з добре відомих прикладів NLP є виявлення спаму в електронній пошті, пошук орфографічних помилок, автоматичні субтитри з перекладом;

в) експертна система – це комп'ютерна система, яка імітує здатність людини приймати рішення. Експертні системи призначені для вирішення складних проблем шляхом міркування за допомогою сукупності знань, представлених в основному у вигляді правил «якщо-то», а не за допомогою звичайного процедурного коду. Експертні системи мають конкретні знання в одній проблемної області, наприклад, медицині, науці, техніці і т. п. Знання експерта називаються базою знань, і вони містять накопичений досвід, який був завантажений і протестований в системі. Подібно до інших систем штучного інтелекту, знання експертної системи можуть бути доповнені доповненнями до бази знань або доповненнями до правил. Чим більше досвіду введено в експертну систему, тим більш якісний результат система може запропонувати;

г) комп'ютерний зір – це наукова область, яка займається дослідженням того, як комп'ютери можуть отримувати та обробляти інформацію високого рівня з цифрових зображень або відео. Метою комп’ютерного зору є зрозуміння та автоматизація завдань, які може виконувати зір людини. Завданнями комп'ютерного зору є способи отримання, обробки,

аналізу і розуміння цифрових зображень і вилучення багатовимірних даних з реального світу для отримання числової або символічної інформації. Розуміння в цьому контексті означає перетворення візуальних образів для опису світу, які мають сенс для розумових процесів і можуть викликати відповідні реакції. Таке розуміння зображення можна розглядати як відділення символічної інформації від даних зображення з використанням моделей, побудованих за допомогою геометрії, фізики, статистики та машинного навчання. Наукова дисципліна комп'ютерного зору пов'язана з теорією штучних систем, які витягують інформацію з зображень. Дані зображення можуть приймати різні форми, такі, як відео з камер, багатовимірні дані з 3D-сканера або медичного скануючого пристрою. Підтеми комп'ютерного зору включають виявлення подій, відстеження об’єктів на відео, розпізнавання об'єктів, оцінку тривимірних зображень та об’єктів, навчання, індексацію, оцінку руху, візуальне обслуговування, тривимірне моделювання сцени і відновлення зображення. Комп’ютерний зір знаходить і аналізує візуальну інформацію за допомогою камери, аналого-цифрового перетворення та цифрової обробки сигналів. Загальний процес комп’ютерного зору включає в себе етап планування деталей, формування вимог і проекту, а потім створення рішення. Під час виконання процес починається з обробки зображень, після чого виконується автоматичний аналіз зображення і вилучення необхідної інформації;

д) робототехніка – це область машинобудування, сфокусована на розробці і виробництві роботів. Роботи часто використовуються для виконання завдань, які людям важко виконувати або виконувати послідовно. Прикладами можуть служити лінії складання автомобілів, в лікарнях, прибирання офісів, роздача продуктів і приготування їжі в готелях, патрулювання фермерських ділянок і навіть в якості поліцейських. Нещодавно машинне навчання використовувалося для досягнення

певних хороших результатів у створенні роботів, які взаємодіють в суспільстві;

е) розпізнавання мови – це технологія, яка дозволяє машині розуміти розмовну мову і переводити його у формат, який може зчитати машина. Це щось на кшталт способу спілкування з комп'ютером, і на основі вхідних команд мовою комп'ютер може виконувати певні завдання. Існує деякий програмне забезпечення для розпізнавання мови, яке має обмежений словниковий запас слів і фраз. Це програмне забезпечення вимагає однозначного розмовної мови для розуміння і виконання конкретного завдання. Сьогодні існують різні програми та пристрої, що містять технологію розпізнавання мови, такі як Cortana, віртуальний помічник Siri, Google Assistant, у Amazon це – Alexa та Alisa від Yandex. Нам потрібно навчити нашу систему розпізнавання мови, щоб розуміти нашу мову. Раніше ці системи були призначені тільки для перетворення мови в текст, але тепер існують різні пристрої, які можуть безпосередньо перетворювати мову у команди. Системи розпізнавання мови можуть використовуватися в багатьох галузях, наприклад: у системах управління або навігаційних системах, у промисловості, для голосового набору.

2.1 Огляд алгоритмів пошуку за зображенням

Задача полягає у тому, щоб створити такий агрегатор, який би мав за свою перевагу пошук продукту за зображенням. Для того щоб обрати найбільш вигідний алгоритм для пошуку за зображенням, потрібно урахувати особливості роботи із зображенням продукту. Наприклад на одному зображенні може бути декілька продуктів водночас і нам потрібно буде сфокусуватись не тільки на об’єкті на зображені у цілому, але і на його частинах. Для того, щоб сформувати уявлення про

те, який алгоритм буде пасувати для даного рішення найкраще, розглянемо, який процес пошуку за зображенням у цілому [3].

Алгоритми знаходження об'єктів для задач комп'ютерного зору є одними з найпотужніших інструментів у всьому машинному навчанні і штучному інтелекті. Це алгоритми прийняття рішень, які дозволяють комп'ютерним системам робити висновки про реальний навколишній світ. Без розпізнавання об'єктів створити роботів, які керують об'єктами, автономними транспортними засобами та програмним забезпеченням для класифікації зображень, було б практично неможливо.

Оскільки виявлення об'єктів являє собою набір алгоритмів і методів, які використовують комп'ютерний зір, давайте спочатку почнемо з визначення комп'ютерного зору і установки контексту для нашого дослідження виявлення об'єктів.

У самих основних термінах комп'ютерний зір – це область комп'ютерних наук, яка прагне наділити комп'ютери здатністю витягувати і інтерпретувати високорівневі функції зображень і відео. Інтерпретація функцій високого рівня імітує спосіб, яким люди розпізнають об'єкти. Люди розпізнають об'єкти за допомогою системи обробки інформації. По-перше, границі об'єкта ідентифікуються і зв'язуються разом, щоб сформувати контур об'єкта. Після цього деталі об'єкта заповнюються, потім завдяки цим сегментам можна навчити мережу розпізнавати по цій вибірці, щоб визначити, що це за об'єкт. Інтерпретація високого рівня зводиться до тогу, що багато дрібних елементи, складові зображення, об'єднуються, і об'єкт розпізнається на найвищому рівні обробки. Аналогічним чином, мета методів комп'ютерного зору полягає в тому, щоб дати комп'ютерів можливість досліджувати зображення і розпізнавати частини зображення, які мають певні значення. Таким чином, замість того, щоб вертати інформацію про об'єкт на зображенні на базовому рівні, система комп'ютерного зору може повертати інформацію на високому рівні (рівні, який має значення для людей). Потім комп'ютер може вибрати відповідний курс дій і виконати інші інструкції в залежності від того, до якого об'єкту зображення було віднесено.

Для алгоритму пошуку за зображенням можна виділити такі основні кроки, зображені на рисунку 2.2.

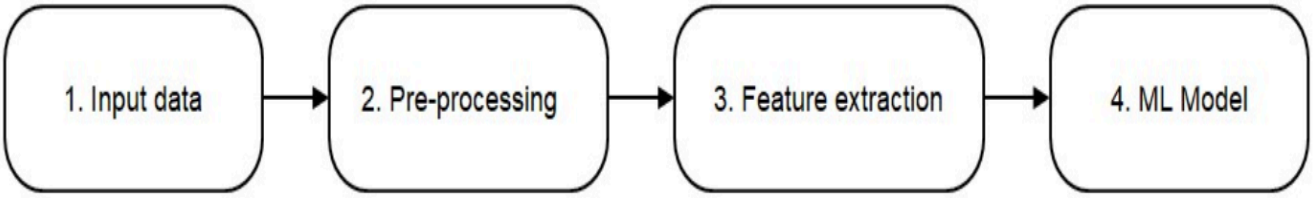

Рисунок 2.2 – Схема кроків розпізнавання об’єкту на зображені

По-перше, можна побачити, що комп’ютер отримує на вхід зображення для аналізу. Зазвичай це фіксується як зображення або послідовність зображень, що утворюють відео.

Кожне зображення проходить через кілька етапів попередньої обробки, метою яких є стандартизація кожного зображення. Поширені етапи попередньої обробки включають розмір зображення, розмивання, зміна повороту, зміну його форми або перетворення зображення з одного кольору в інший – наприклад, від RGB або HSL до монохромної палітри, тобто відтінків сірого. Стандартизація зображення робиться для того, щоб потім було зручніше робити аналіз зображення та результати були більш точними.

Далі ми витягуємо інформацію, яка буде мати характер особливої інформації, з зображення за допомогою сегментації або тільки с використанням ключових точок. Процес виділення ознак – це те, що допомагає нам визначати певні об'єкти, і вони зазвичай є інформацією про форму або колір предмета. Результатом цього процесу є вектор функції, що представляє собою список унікальних фігур, що ідентифікують об'єкт, або сегменти.

Нарешті, ці функції подаються в класифікаційну модель машинного навчання. Цей крок розглядає вектор функцій з попереднього кроку і прогнозує клас зображення. Можна уявити, що є моделлю класифікатора, і переходити до процесу класифікації.

Розглянемо два підходи для виділення ознак на зображені. Перший, це пошук ключових точок на зображенні та дескрипторів, для пошуку ключових точок існують такі алгоритми, як SURF, BRISK, Harris, FAST, MSER. А для знаходження дескрипторів SURF, BRISK, FREAK, Block. А другий – це сегментація зображення.

2.2 Знаходження особливих точок на зображенні

У кожного зображення на якому існують якісь об’єкти є свої особливі точки. Особливими точками є такі точки, за якими можна класифікувати зображення, розпізнати його, якась особливість зображення, виділити його унікальність.

Як правило ці точки знаходяться там, де різко змінюється колір, яскравість, кут. Потрібно вибирати такі точки, які описують, характеризують зображення, також необхідно враховувати, що особливими точками повинні бути такі точки, які з великою ймовірністю будуть знайдені на іншому зображенні для того , щоб мати змогу порівняти ці зображення, як це потрібно реалізувати і для агрегатору пошуку товарів за зображенням.

Кожен метод виявлення особливих точок повинен гарантувати гарантувати інваріантність щодо перетворень зображення. Потрібно дослідити, яким чином комп'ютер розуміє, які ключові точки різних зображень відповідають один одній. Бо у кожної точки на різних зображеннях різні координати, яким чином йде їх зіставлення. Для цього кожній окремій точці необхідно присвоїти опис, яке буде однаковим на різних зображеннях. Цей опис робиться за допомогою дескрипторів.

Дескриптор – це ідентифікатор особливої точки, який робить її унікальною щодо інших особливих точок. Використовуючи дескриптори потрібно пам’ятати про інваріантність перетворення зображень.

Можна представити такий алгоритм пошуку об’єктів на зображенні:

а) на зображенні знаходяться особливі точки і їх дескриптори;

б) за допомогою дескрипторів серед цих точок виявляються пари відповідних точок;

в) по цим парам йде побудова моделі перетворення зображення.

Нижче наведено декілька перетворень, щодо яких потрібно отримати інваріантність:

а) зміна яскравості;

б) масштабування;

в) обертання;

г) зсув.

Розглянемо більш детально алгоритми пошуку особливих точок.

Алгоритм знаходження точок SURF працює шляхом виявлення особливих точок за допомогою Гессіана (матриці Гессе). Потім для точок із найбільшою яскравістю вираховується значення напрямку яскравості. Далі формуються дескриптори. Для методу SURF дескриптор складеться із 64 чисел. Матриця Гессе має інваріантність щодо обертання зображень. Але вона інваріантно вимірює масштабування [4]. Цей метод SURF використовує фільтри різних масштабів для розрахунку Гессіану. потім для кожного окремого пункту вираховується градієнт і масштаб. Градієнт у точках обчислюється за допомогою Хаара.

Після того, як були знайдені особистісні точки методу SURF, обчислюються їх дескриптори.

Кожне число відображає різницю градієнта навколо особливої точки. Кожна особлива точка являє собою максимум Гессіан, що гарантує факт того, що в околиці цієї точки будуть ділянки з різними градієнтами. Таким чином, забезпечується відмінність дескрипторів для різних особливих точок, внаслідок чого з’являється інваріантність дескриптора щодо обертання.

Основною відмінністю методу від методу BRISK від методу SURF є те, що метод оцінює істинний масштаб кожної ключової точки в безперервному масштабі простору.

Для круговим околицях потенційних ключових точок застосовується Гаусове розмивання. Для визначення напрямку ключової точки використовується сума локальних градієнтів.

За результатами досліджень, одним з найоптимальніших детекторів кутів є популярний метод Харріса та алгоритм знаходження точок Harris. Даний метод – поліпшення методу Моравіца, автори розглядають похідні зміни яскравості зображення для дослідження змін яскравості в усіх напрямках. Метод Харріса інваріантний до обертання, а також до змін яскравості. Однак він чутливий до шумів і залежить від масштабу зображення (в іншому випадку використовують багатомасштабний метод Харріса.

Вищеописані методи визначають особливі точки, застосовуючи свої алгоритми безпосередньо до пікселів вихідного зображення. Існує альтернативний підхід, що полягає в використанні алгоритмів машинного навчання для тренування класифікатора точок на деякій множині зображень.

Метод та алгоритм знаходження точок FAST будує дерева рішень для класифікації пікселів [5]. Для кожного пікселя p розглядається коло радіусом 4, вписане в квадратну область зі стороною 7 пікселів (рис.2.3).

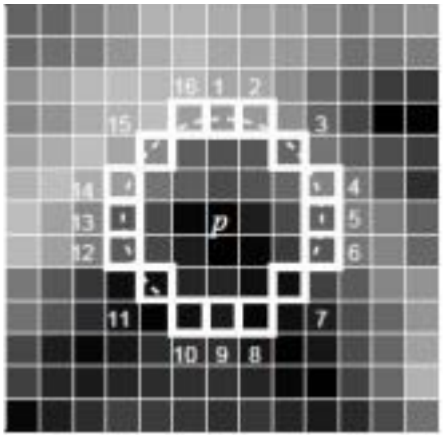

Рисунок 2.3 – Робоча околиця пікселя при використанні FAST детектора .

Окружність проходить через 16 пікселів. Кожен з пікселів кола відносно пікселя p може перебувати в одному з 3х станів: Для кожного знайденого x і

розділимо безліч на 3 підмножини точок, які темніше, схожі або світліше точки x відповідно. Потім будується дерево рішень.

За результатами побудованого дерева визначаються кути на тестових зображеннях.

Метод та алгоритм знаходження точок MSER вирішує проблему інваріантності особливих точок при різних масштабах зображення. Виділяються кілька різних регіонів з екстремальними властивостями функції інтенсивності як всередині регіону, так і на її кордонах.

Алгоритм розглядається для чорно-білого зображення. У підсумку виходить набір бінарних зображень при різних значеннях деякого порога значень (піксель, значення якого менше цього порога вважається чорним, у іншому випадку – білим). Далі будується піраміда, у якій при мінімальному пороговому значенні виходить біле зображення, при максимальному – чорне. У разі, якщо помічено будь-який рух, на білому зображенні з'являються чорні плями, які є локальними мінімумами інтенсивності [5]. При збільшенні порога чорні плями починають збільшуватися і зливатися, в кінцевому підсумку утворюючи повністю чорне зображення. Така піраміда дозволяє побудувати безліч зв'язкових компонент, відповідних білим областям – регіонах з максимальним значенням інтенсивності. Відповідно, при інвертуванні зображення отримаємо чорні плями – регіони з мінімальним значенням інтенсивності.

Алгоритм складається з декількох етапів:

- сортувати безліч всіх пікселів зображення в порядку зростання або убування інтенсивності;
- виконується побудова піраміди зв'язкових компонент. Для кожного пікселя відсортованого безлічі виконується послідовність дій: виконується пошук локальних мінімумів для всіх компонент, тобто пошук пікселів, присутніх в даній компоненті, але не входять до складу попередніх). Набір локальних мінімумів рівня відповідає екстремального регіону на зображенні.

Алгоритм побудови дескриптора SURF має наступні кроки:

- навколо точки будується квадратна околиця розміром 20s, де s – масштаб, на якому отримано максимальне значення детермінанта матриці Гессе;
- отримана квадратна область розбивається на блоки, в результаті область буде розбита на 4x4 сегменти, приклад розбиття можна побачити на рисунку 2.4

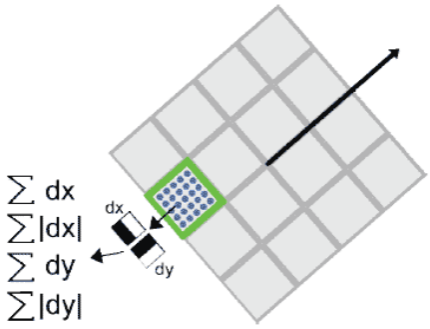

Рисунок 2.4 – Область розбиття на 4x4

- для кожного блоку обчислюються більш прості ознаки. Як наслідок, виходить вектор, що містить 4 компоненти, де 2 – це сумарний градієнт по квадранту, 2 – сума модулів точкових градієнтів;
- дескриптор формується в результаті склеювання зважених описів градієнта для 16 квадрантів навколо особливої точки. Елементи 14 дескриптора зважуються на коефіцієнти Гауссова ядра. Ваги необхідні для більшої стійкості до шумів у віддалених точках;
- додатково до дескриптора заноситься слід матриці Гессе. Ці компоненти необхідні, щоб розрізняти темні і світлі плями. Для світлих точок на темному тлі слід негативний, для темних точок на світлому фоні – позитивний.

Метод SURF також використовується для пошуку об'єктів. Проте, дескриптор ніяк не використовує інформацію про об'єкти. SURF розглядає

зображення як єдине ціле і виділяє особливості всього зображення, тому він погано працює з об'єктами простої форми [6].

Дескриптор Block. Алгоритм дуже простий: навколо особливих точок береться околиця з заданим радіусом (зазвичай береться радіус у 11 пікселів), потім формується дескриптор, що складається з безлічі чисел, які є параметрами яскравості пікселів.

2.3 Сегментація зображень

Сегментацію зображення можна визначити як розподілення всіх об’єктів зображення або пікселів на зображенні на різні кластери, які демонструють подібні особливості. Сегментація включає поділ зображення на групи пікселів, які є рівномірними щодо певної норми.

Розділення зображень розглядається як необхідна фундаментальна операція для важливого дослідження та розуміння отриманої картини. Це основна і фундаментальна частина дослідження зображення, або потенційно розроблена концепція розпізнавання, і є одним з найпотужніших починань з підготовки зображень, що визначає характер останнього поділу.

Розглянемо діаграму класифікації алгоритмів для сегментації зображень на рисунку 2.5.

Регіональна сегментація (Region Based) або сегментація на основі регіону – це метод визначення регіону безпосередньо.

Зростання регіонів – це простий метод сегментації зображень на основі регіону. Він також класифікується як піксельний метод сегментації зображень, оскільки він включає вибір початкових точок насіння. Початковий набір невеликих площ ітераційно зливаються відповідно до обмежень подібності. Обирається довільний піксель та порівнюється із сусідніми пікселями. Область вирощується з пікселя шляхом додавання до сусідніх пікселів які подібні, збільшуючи розмір

регіону. Коли зростання одного регіону припиняється, ми просто вибираємо інший піксель, який ще не належить до жодного регіону, і увесь процес починається заново.

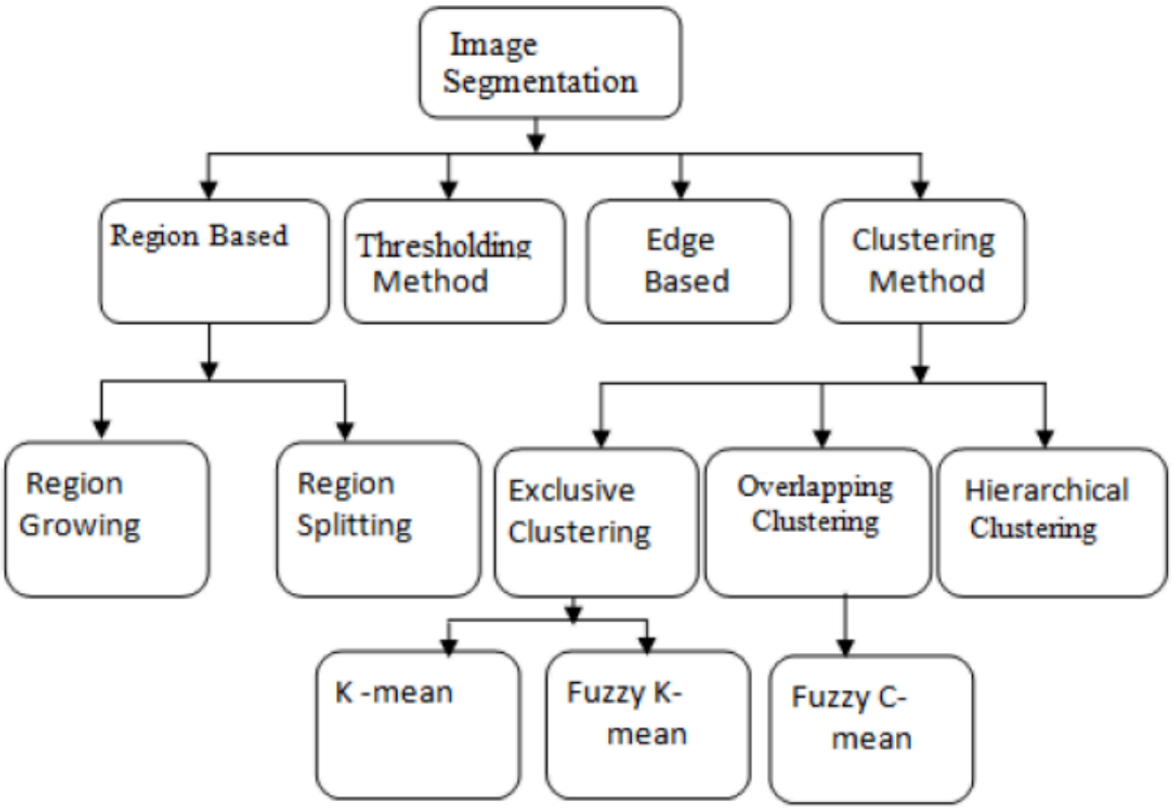

Рисунок 2.5 – Класифікація алгоритмів сегментації зображень

Метод розщеплення регіону – це протилежністю методу росту регіону. Ця техніка працює на повному зображенні. Розщеплення регіонів – це підхід зверху вниз. Він працює з повним зображенням і розбиває його так, щоб відокремлених частин було більше особливих, ніж загальних, це нагадує деревоподібну структуру. Зображення послідовно поділяється на маленькі сегменти на основі критерію однорідності, і аналогічні області об'єднуються для отримання сегментованого результату.

Порогування (Thresholding Method) або поріг зображення – це простий, але ефективний спосіб розподілу зображення на передній план та фон. Ця методика

аналізу зображень – це тип сегментації зображень, який ізолює об'єкти шляхом перетворення зображень у масштабах сірого у бінарні зображення.

Виявлення контурів (Edge Based)– це процес пошуку краю зображення. Виявлення контурів на зображенні є дуже важливим кроком на шляху до аналізу особливостей зображення. Краї складаються із особливих рис і містив важливу інформацію. Це значно знижує розмір зображення і фільтрує інформацію, яка може розглядатись, як менш актуальна, зберігаючи важливі структурні властивості зображення. Оскільки краї часто виникають на місцях, що представляють межі об'єкта, виявлення контурів використовується для сегментації зображень, коли зображення розділені на ділянки, що відповідають різним об’єктам [7].

Сегментація з використанням методів кластеризації. Для того, щоб виконати кластеризацію для сегментації зображень, нам спочатку потрібно вибрати простір для зображення, а потім використовувати відповідний захід розділення (міра близькості), координувати зображення та групові фокуси у вибраному просторі зображень. Для задачі розпізнавання товарів у кластеризація має свої переваги. По-перше, таке рішення добре масштабується, по-друге, потрібно враховувати особливості інтернет маркет плейсів, наприклад на фото з продажу одягу може бути кілька одиниць товарів і нам потрібно буде зробити пошук за товаром, який може бути на даному зображені, але не буде відноситись до лоту. Таким прикладом можна назвати наявність на фото сорочки та капелюха на моделі для лоту капелюха, але при пошуковому запросі об’єктом для пошуку була сорочка, то є вірогідність, що алгоритм при пошуку запропонує лот з капелюхом, бо на фото присутня сорочка [8]. Щоб уникнути таких випадків, дуже зручно порівнювати кластери, що отримуються в результаті сегментації із класами, що створюються при роботі парсеру.

3 АГОРИТМИ КЛАСТЕРІЗАЦІЇ

3.1 Алгоритм k-найближчих сусідів.

Дуже простий і дуже ефективний. Прогнози робляться для нової точки даних шляхом пошуку всього навчального набору для K-подібних екземплярів (сусідів) та узагальнення вихідної змінної для цих екземплярів K. Найбільший випадок використання k-Найближчих сусідів - це системи рекомендацій, в яких, якщо ми знаємо, що користувачеві подобається певний предмет, ми можемо рекомендувати подібні елементи для них [9].

Передбачається, що вже є якась кількість об'єктів з точною класифікацією (тобто для кожного них точно відомо, якого класу він належить). Потрібно виробити правило, що дозволяє віднести новий об'єкт до одного з можливих класів (тобто самі класи відомі заздалегідь).

В основі k-NN лежить таке правило: об'єкт вважається належним того класу, до якого належить більшість його найближчих сусідів. Під «сусідами» тут розуміються об'єкти, близькі до досліджуваного в тому чи іншому сенсі. Зауважимо, що тут необхідно вміти визначати, наскільки об'єкти близькі один до одного, тобто вміти вимірювати «відстань» між об'єктами. Це не обов'язково евклідова відстань. Це може бути міра близькості об'єктів, наприклад, за кольором, формою, матеріалом і т.п. Отже, для застосування методу k-NN в просторі ознак об'єктів повинна бути введена деяка метрика, тобто функція відстані.

Передбачається, що об'єкти з близькими значеннями одних ознак будуть близькі і за іншими ознаками (тобто стосуватися одного і того ж класу).

В процесі навчання алгоритм просто запам'ятовує всі вектори ознак і відповідні їм мітки класів. При роботі з реальними даними, тобто спостереженнями, мітки класу яких невідомі, обчислюється відстань між вектором нового спостереження і раніше після успішної реєстрації. потім вибирається k найближчих до нього векторів, і новий об'єкт відноситься до класу, якому належить більшість з них, візуалізацію кластеру можна побачити на рисунку 3.1.

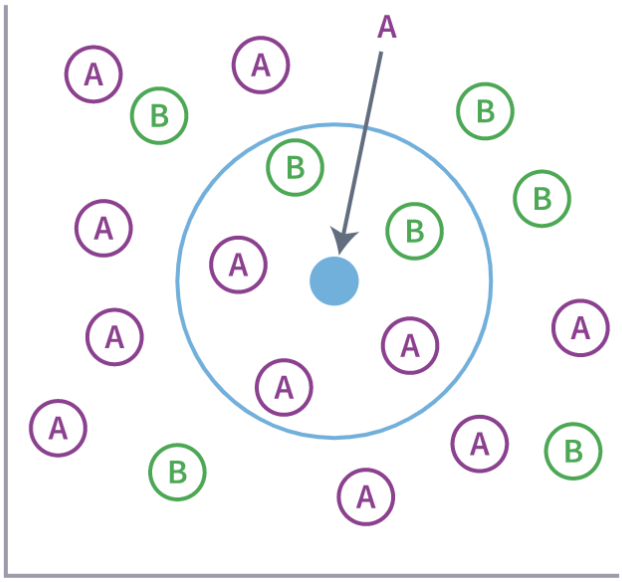

Рисунок 3.1 – k-найближчих сусідів

Вибір параметра k суперечливий. З одного боку, збільшення його значення підвищує достовірність класифікації, але при цьому кордону між класами стають менш чіткими [10].

На практиці хороші результати дають евристичні методи вибору параметра k, Наприклад, перехресна перевірка. Незважаючи на свою відносну алгоритмічну простоту метод показує гарні результати. Головним його недоліком є висока обчислювальна трудомісткість, яка збільшується квадратично з ростом числа навчальних прикладів.

3.2 Дерево рішень

Дерева рішень це ще один важливий тип методики класифікації, що використовується для машинного навчання прогнозування, але не дуже пасує для кластеризації зображень.

Будуються на основі навчання з учителем. В якості навчального набору даних використовується множина спостережень, для яких попередньо задана мітка класу.

Структурно дерево рішень складається з об'єктів двох типів – вузлів (node) і листя (leaf). У вузлах розташовані вирішальні правила і підмножини спостережень, які їм задовольняють. У листі містяться класифіковані деревом спостереження: кожен лист асоціюється з одним з класів, і об'єкту, який розподіляється в лист, присвоюється відповідна позначка класу. дерево рішень.

Візуально вузли і листя в дереві добре помітні: в вузлах вказуються правила, що розбивають що містяться в ньому спостереження, і проводиться подальше розгалуження. У листі правил немає, вони позначаються міткою класу, об'єкти якого потрапили в даний лист. Розгалуження в листі не проводиться, і вони закінчують собою гілку дерева (тому їх іноді називають термінальними вузлами). Якщо клас, присвоєний деревом, збігається з цільовим класом, то об'єкт є розпізнаним, в іншому випадку – нерозпізнаним. Самий верхній вузол дерева називається кореневим (root node). У ньому міститься весь навчальний або робочий набір даних.

Дерево рішень є лінійним класифікатором, тобто виробляє розбиття об'єктів в багатовимірному просторі площинами (в двовимірному випадку – лініями). Дерево рішень як лінійний класифікатор Дерево, представлене на рисунку 3.2, вирішує завдання класифікації об'єктів за двома атрибутам на три класи. На малюнку гуртки представляють об'єкти класу 1, квадрати - класу 2, а трикутники - класу 3.

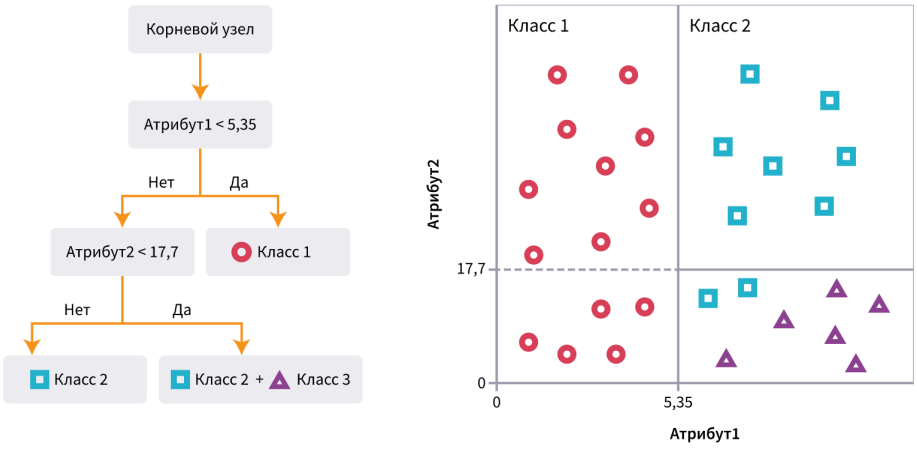

Рисунок 3.2 – Дерево рішень

Простір ознак розділене лініями на три підмножини, асоційованих з класами. Ці ж підмножини будуть відповідати і трьом можливим наслідків класифікації. У класі «трикутників» є нерозпізнані приклади, тобто приклади, що потрапили в підмножини, асоційовані з іншим класом [11].

Теоретично, алгоритм може генерувати нові розбиття до тих пір, поки всі приклади не будуть розпізнані правильно, тобто поки підмножини, асоційовані з листям, не стануть однорідними за класовим складом. Однак це призводить до ускладнення дерева: велике число розгалужень, вузлів і листя ускладнює його структуру і погіршує його реалізацію. Тому на практиці розмір дерева обмежують навіть за рахунок деякої втрати точності. Даний процес називається спрощенням дерев рішень і може бути реалізований за допомогою методів ранньої зупинки і відсікання гілок.

3.3 Наївний Баєсів класифікатор

Наївний Баєсів класифікатор - класифікатор, що використовує теорему Баєса для визначення ймовірності приналежності спостереження (елемента вибірки) до одного з класів при припущенні (наївному) незалежності змінних. Тобто, якщо на основі значень змінних можна однозначно визначити, до якого класу належить спостереження, Байєсів класифікатор повідомить ймовірність приналежності до цього класу. У проміжних же випадках, коли спостереження може з різною ймовірністю належати до різних класів, результатом роботи класифікатора буде вектор, компоненти якого є ймовірностями приналежності до того чи іншого класу.

Можна бачити, що ідеальний байєсів класифікатор в якомусь сенсі є оптимальним. Його результат не може бути поліпшений, тому що у всіх випадках, коли можлива однозначна відповідь, він його дасть - а в тих випадках, коли відповідь неоднозначна, результат кількісно характеризує міру цієї неоднозначності. Разом з тим, в оптимальності криється і основний недолік

ідеального байєсового класифікатора: для його побудови потрібна вибірка, що містить всі можливі комбінації змінних - а розмір такої вибірки експоненціально зростає із зростанням числа змінних [12].

Для подолання описаної вище проблеми на практиці використовують наївний байєсів класифікатор - класифікатор, побудований на основі припущення про незалежність змінних, тобто припущення про те, що використання цього припущення дозволяє не вивчати взаємодію всіх можливих поєднань змінних, обмежившись лише впливом кожної змінної окремо на приналежність образу до одного з класів. Перевагою цього підходу є те, що вимоги до розміру вибірки скорочуються від експоненційних до лінійних.

Недоліком є те, що модель є точною лише у випадку, коли виконується припущення про незалежність. В іншому випадку, строго кажучи, обчислені ймовірності вже не є точними (і навіть більше того, їх сума може не дорівнювати одиниці, через що потрібно нормувати результат). Однак на практиці незначні відхилення від незалежності призводять лише до незначного зниження точності, і навіть у разі істотної залежності між змінними результат роботи класифікатора продовжує корелювати з істинною приналежністю образу до класів. При цьому переваги класифікатора (висока швидкість роботи, простота і масштабованість, помірні вимоги до пам'яті) часто переважують недоліки [13].

3.4 Алгоритм k-середніх

Метод k-середніх використовується для кластеризації даних на основі алгоритму розбиття векторного простору на заздалегідь визначену кількість кластерів k.

Алгоритм являє собою ітераційну процедуру, в якій виконуються наступні кроки. Вибирається число кластерів k. З вихідної множини даних випадковим чином вибираються k спостережень, які будуть служити початковими центрами

кластерів. Для кожного спостереження вихідного безлічі визначається найближчий до неї центр кластера (відстані вимірюються в метриці Евкліда). При цьому записи, «притягнуті» певним центром, утворюють початкові кластери. Обчислюються центроїди - центри ваги кластерів. Кожен центр ваги - це вектор, елементи якого являють собою середні значення відповідних ознак, обчислені за всіма записами кластера. Центр кластера зміщується в його центр ваги, після чого центр ваги стає центром нового кластера. 3-й і 4-й кроки повторюються [14]. На кожній ітерації відбувається зміна меж кластерів і зміщення їх центрів. В результаті мінімізується відстань між елементами всередині кластерів, і збільшуються міжкластерні відстані (рисунок 3.3). Зупинка алгоритму проводиться тоді, коли кордони кластерів і розташування центроїдів не перестануть змінюватися від ітерації до ітерації, тобто на кожній ітерації в кожному кластері буде залишатися один і той же набір спостережень. На практиці алгоритм зазвичай знаходить набір стабільних кластерів за кілька десятків ітерацій.

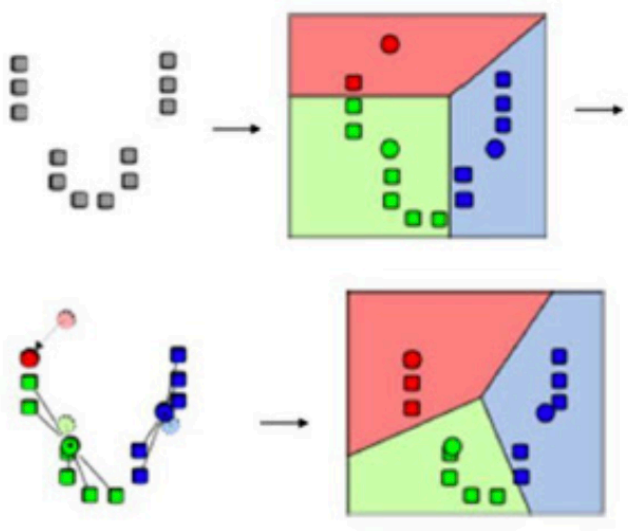

Рисунок 3.3 – Формування кластерів

Перевагою алгоритму є швидкість і простота реалізації.

До недоліків можна віднести невизначеність вибору початкових центрів кластерів, а також те, що він погано працює на зашумлених зображеннях.

3.5 Нечітка кластеризація (Fuzzy C-means)

Найбільш захопливим алгоритмом нечіткого кластеризації є алгоритм нечіткої кластеризації C-means. Це метод кластеризації, який дозволяє одночасно належати одному фрагменту даних належать до двох або більше кластерів. Наприклад, точка даних, що знаходиться близько до центру кластера, матиме високий ступінь належності в цьому кластері, а інша точка даних, що знаходиться далеко від центру кластера, матиме низький ступінь приналежності до цього кластеру .

Кластеризація C-means, починається з рандомної початкової спроби пошуку центру кластерів; це середнє місце розташування кожного кластеру [15]. Потім призначає кожній точці даних випадкову оцінку належносі для кожного кластеру. Ітераційним оновленням центрів кластерів та оцінок членства для кожної точки даних переміщує центри кластерів у правильне місце розташування в наборі даних і для кожної точки даних знаходить ступінь належності до кожного кластеру. Ця ітерація мінімізує функцію, яка представляє відстань від будь-якої заданої точки даних до центру кластера, оцінене на приналежність цієї точки даних у кластері.

3.6 Ієрархічна кластеризація

Ідея ієрархічної кластеризації полягає у розробці дерева кластерів (для зображення, відомих як пікселі) та рівнями порівнянності, на яких групування змінюється. Ієрархічна кластеризація включає створення кластерів, які мають заздалегідь визначений упорядкування зверху вниз. Наприклад, усі файли та папки на жорсткому диску організовані в ієрархії. Існує два типи ієрархічної кластеризації – метод поділу та агломераційна [16].

У методі поділу, ми присвоюємо всі спостереження одному кластеру, а потім розділяємо кластер на два найменш схожі кластери. Потім продовжуємо рекурсивно на кожному кластері, поки не буде одного кластеру для кожного спостереження. Існує думка, що алгоритми поділу, створюють більш точні ієрархії, ніж агломераційні алгоритми в деяких обставинах, але більш складні.

При агломераційному або кластерному методі знизу вгору ми присвоюємо кожне спостереження своєму кластеру. Потім обчислюється подібність (наприклад, відстань) між кожним кластером і з’єднуються два найбільш схожі кластери. Потім повторюються кроки поки не залишиться лише один кластер.

4 МОДЕЛЮВАННЯ СИСТЕМИ ТА ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ

4.1 Створення парсеру та збір даних

Для того, щоб реалізувати агрегатор для пошуку товарів з функцією пошуку за зображення, спершу нам потрібно отримати дані з декількох джерел інформації. Для цього необхідно реалізувати парсер або синтаксичний аналізатор [20-23].

Синтаксичний аналізатор – це програма, або її частина, яка приводить дані або до певної структури.

В нашому випадку текст – це HTML-код веб-сторінки або дані у форматі JSON.

Для реалізації подібного продукту потрібно створити веб-сервер, який зможе скачувати вихідний формат веб-сторінки певного сайту та аналізувати її.

Для цього пропонується обрати Node.js у якості серверного фреймворку.

Принцип роботи парсеру:

а) ітеративно обійти усі пошукові сторінки веб-сайту, які було обрано для аналізу даних;

б) отримати посилання на сторінки конкретних товарів;

в) скачати код сторінки товару;

г) відокремити потрібні дані зі сторінки;

д) зберегти їх у базу даних.

Спершу потрібно визначити URL сторінки. Візьмемо для прикладу сайт магазину ASOS.

За допомогою експериментів зі сторінкою: можна зробити висновок, що для навігації по пошуку необхідно змінювати параметр «page={номер_сторінки }».

Таким чином, зробивши певну кількість запитів, можна отримати повний перелік усіх товарів за фільтром. Оскільки кожний сайт унікальний, а також товари не знаходяться в одній категорії, необхідно перед стартом парсера зробити таку процедуру вручну, визначивши цікаві нам категорії товарів.

Далі нам потрібно надати парсеру інструкції, де саме знаходяться посилання на індивідуальні сторінки товарів. Для цього потрібно скористатися інспектором браузера та обрати селектори тегів, у яких знаходиться інформація, що потрібна для збору, приклад аналізу даних зображено на рисунку 4.1.

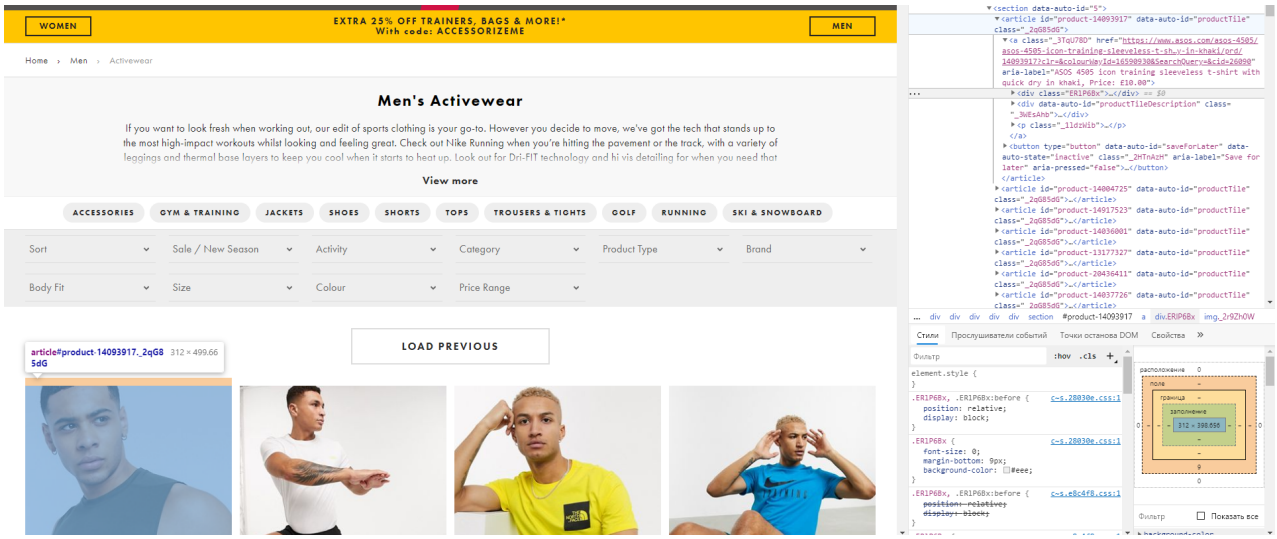

Рисунок 4.1 – Пошук потрібного елемента у DOM

За допомогою css-селекторів дістаємось до потрібного тега.

У випадку, зазначеному вище шлях буде наступним: «div[data-auto-id=”productList”] article > a». Ітеративно потрібно отримати усі посилання на окремі сторінки з усіх сторінок пошуку.

У результаті ми отримаємо масив посилань на товари. Таким же чином скачуємо окремі сторінки.

Парсер скачає сторінку у HTML-форматі, приклад вхідних даних можна побачити на рисунку 4.2. За допомогою інспектора шукаємо css-шлях до елементів, що нас цікавлять. В даному випадку це ціна, назва, зображення, опис та посилання на ресурс, з якого було зібрано дан.

Далі необхідно поставити у відповідність селектори та поля у базі даних. Після цього треба зберегти отримані дані.

Для реалізації подібного парсеру є декілька шляхів або їх комбінація. Розглянемо типові проблеми та варіанти їх подолання.

Оскільки кожен сайт є унікальним, принципи роботи парсера можуть змінюватися від одного ресурсу до іншого.

```
<!DOCTYPE html>
<!--11.0.12489--><html lang="en-GB" data-platform="desktop" data-store-code="ROW" data-country-code="UA">
<head><meta charSet="utf-8"/><title>Women's Tops Sale | Tops For Sale | ASOS</title><meta name="description" cc
latest collection of tops for women and shop your favourite items on sale. Pick yours and order at ASOS."/><meta name
<meta name="msvalidate.01" content="0BC251BA275FCFB5D0505FEBC89F03B6"/><meta property="fb:app_id" content="1230736010
initial-scale=1"/><meta name="Slurp" content="NOODP"/><meta name="Slurp" content="NOYDIR"/><link rel="apple-touch-icc
rel="icon" type="image/png" sizes="32x32" href="/favicon-32x32.png"/><link rel="icon" type="image/png" sizes="16x16"
href="/manifest.json"/><link rel="mask-icon" href="/safari-pinned-tab.svg" color="#2d2d2d"/><link href="https://asset
href="https://www.asos.com/api" rel="preconnect"/><link href="https://my.asos.com/my-account" rel="preconnect"/><meta
    <link rel="alternate" hreflang="en-GB" href="https://www.asos.com/women/sale/tops/cat/?cid&#x3D;4167" />
    <link rel="alternate" hreflang="en-US" href="https://www.asos.com/us/women/sale/tops/cat/?cid&#x3D;4167" />
    <link rel="alternate" hreflang="fr-FR" href="https://www.asos.com/fr/femme/promos/tops/cat/?cid&#x3D;4167" />
    <link rel="alternate" hreflang="de-DE" href="https://www.asos.com/de/damen/sale/oberteile/cat/?cid&#x3D;4167" />
    <link rel="alternate" hreflang="it-IT" href="https://www.asos.com/it/donna/saldi/top/cat/?cid&#x3D;4167" />
    <link rel="alternate" hreflang="es-ES" href="https://www.asos.com/es/mujer/rebajas/tops/cat/?cid&#x3D;4167" />
    <link rel="alternate" hreflang="en-AU" href="https://www.asos.com/au/women/sale/tops/cat/?cid&#x3D;4167" />
    <link rel="alternate" hreflang="ru-RU" href="https://www.asos.com/ru/women/rasprodazha/topy/cat/?cid&#x3D;4167" /
    <link rel="alternate" hreflang="x-default" href="https://www.asos.com/women/sale/tops/cat/?cid&#x3D;4167" />
    <link rel="alternate" hreflang="sv-SE" href="https://www.asos.com/se/kvinna/rea/toppar/cat/?cid&#x3D;4167" />
    <link rel="alternate" hreflang="nl-NL" href="https://www.asos.com/nl/dames/sale/tops/cat/?cid&#x3D;4167" />
    <link rel="alternate" hreflang="da-DK" href="https://www.asos.com/dk/kvinder/udsalg/toppe/cat/?cid&#x3D;4167" />
    <link rel="alternate" hreflang="pl-PL" href="https://www.asos.com/pl/kobiety/wyprzedaz/topy/cat/?cid&#x3D;4167" /
<link rel="canonical" href="https://www.asos.com/women/sale/tops/cat/?cid&#x3D;4167"/><script src="https://assets.asc
<script src="https://assets.asosservices.com/asos-ui/manifest/featuresManifest.js"></script><script
src="https://assets.asosservices.com/sitechromepublisher//dist/client.newRelicPro.e01db7.js"></script><script>NREUM.i
data.net",licenseKey:"3edea625b0",applicationID:"46901220",sa:1}</script><script type="text/javascript"
src="https://assets.asosservices.com/MasterLayout/Javascript/MasterLayout/core.0.4.3.min.js"></script><script src="ht
</script><link href="https://assets.asosservices.com/sitechromepublisher//dist/client.sitechrome.a89ab59e2b637d18e2b9
href="https://assets.asosservices.com/plp/plp/styles/c~s.e56d7a.css" media="screen, projection" rel="stylesheet" type
href="https://assets.asosservices.com/plp/plp/styles/c.e56d7a.css" media="screen, projection" rel="stylesheet" type="
href="https://assets.asosservices.com/plp/plp/styles/3_plp.e56d7a.css" media="screen, projection" rel="stylesheet" ty

<link rel="next" href="https://www.asos.com/women/sale/tops/cat/?cid=4167&page=2" />
<meta property="og:url" content="https://www.asos.com/women/sale/tops/cat/?cid=4167" />
<meta property="og:type" content="product.group" />
<meta property="og:title" content="Women's Tops Sale | Tops For Sale | ASOS" />
<meta property="og:description" content="Discover women's tops on sale at ASOS. Shop the latest collection of tops fc
order at ASOS." />
<meta property="og:image" content="https://assets.asosservices.com/asos-ui/favicons/social-share-1x.jpg" />
<meta name="twitter:card" content="summary" />
<meta property="twitter:site" content="@asos" /><link rel="preload" href="https://assets.asosservices.com/MasterLayou
crossOrigin="anonymous"/><link rel="preload" href="https://assets.asosservices.com/MasterLayout/WebFonts/FTN75__W/FTN
crossOrigin="anonymous"/>
  <script>!function(e){var n="https://s.go-mpulse.net/boomerang/";if("True"=="True")e.BOOMR_config=e.BOOMR_config||{}
{},e.BOOMR_config.PageParams.pci=!0,n="https://s2.go-mpulse.net/boomerang/";if(window.BOOMR_API_key="KEKMC-5QPEH-Y5FH
e=document.createElement("script");e.id="boomr-scr-as",e.src=window.BOOMR.url,e.async=!0,i.parentNode.appendChild(e),
n,t,a,r,d=document,O=window;if(window.BOOMR.snippetMethod=e?"if":"i",t=function(e,n){var t=d.createElement("script");
Date).getTime(),e=e||d.body,e.appendChild(t)},!window.addEventListener&&window.attachEvent&&navigator.userAgent.match
t(i.parentNode,"boomr-async");a=document.createElement("IFRAME"),a.src="about:blank",a.title="",a.role="presentation"
(a.frameElement||a).style,r.width=0,r.height=0,r.border=0,r.display="none",i.parentNode.appendChild(a);try{O=a.conten
{n=document.domain,a.src="javascript:var d=document.open();d.domain='"+n+"';void(0);",O=a.contentWindow,d=O.document.
<bo"+"dy onload='document._boomrl();'>");else if(O._boomrl=function(){t()},O.addEventListener)O.addEventListener("loa
if(O.attachEvent)O.attachEvent("onload",O._boomrl);d.close()}function a(e){window.BOOMR_onload=e&&e.timeStamp||(new
Date).getTime()}if(!window.BOOMR||!window.BOOMR.version&&!window.BOOMR.snippetExecuted){window.BOOMR=window.BOOMR||{}
```


Рисунок 4.2 – Приклад вхідних даних

В загальному, парсер повинен приймати на вхід конфігурацію, складовими якої повинні бути:

– вхідний URL зі змінюваним параметром номера сторінки;

– конфігурація для отримання посилань на окремі товари;

– конфігурація для отримання даних зі сторінки товару.

У самому простому випадку цього буде достатньо для роботи з багатьма сайтами.

Є проблеми при створені парсеру та його роботи.

По-перше, це single-page додатки, де у тексті HTML не містяться дані. Вони надходять пізніше та рендеряться із затримкою. Щоб подолати цю проблему пропонується використовувати сервіс Puppeteer. Puppeteer – це бразуер Chrome для серверу, який не має графічної оболонки. За допомогою нього можна завантажити сторінку, і лише після повного відображення виокремити дані, які нас цікавлять. Реалізацію методу для парсингу single-page додатка для можна побачити нижче на рисунку 4.3.

```
async function downloadSinglePage(url){
    const optionsCommon = {
        uri: url,
        headers: {cookie: 'geocountry=UA; check=true; AMCVS_C0137F6A52DEAFCC0A490D4C%40AdobeOrg=1; browseCountry=UA; browseCurrency=GBP
        transform: function (body) {
            return cheerio.load(body);
        }
    };

    let $common;
    try {
        $common = await request(optionsCommon);
    } catch (err) {
        console.error(err)
    }

    const price = $common('div.product-price span').text();
    const title = $common('div.product-hero h1').text();
    const description = $common('div.product-description ul').text();
    const photos = $common('div.thumbnails ul li button img');
    const parsedPhotos = [];
    for (const prop in photos) {
        if (!isNaN(parseInt(prop))) {
            parsedPhotos.push(photos[prop].attribs.src.replace('$S$', '$XXL$').replace( searchValue: 'wid=40',  replaceValue: 'wid=513'));
        }
    }
    const type = $common('div.product-description a:first-child').text();


    const base64 = [];
    for (let i = 0; i < parsedPhotos.length; i++) {
        const res = await axios(parsedPhotos[i]);

        const buff = new Buffer(res.data);
        const base64data = buff.toString('base64');

        base64.push(base64data);
    }

    return {title, url, description, type, photos: parsedPhotos, price, base64Photos: base64};
}
```

Рисунок 4.3 – Реалізація парсингу для Single-page застосунків

По-друге, це захист від DDOS-атак. В такому випадку наш парсер стикнеться з проблемою відмови у запиті. В свою чергу є декілька способів, як подолати цю проблему. В першу чергу можна робити запити з інтервалом. Таким чином для

сервера-відповідача це буде виглядати, наче користувач просто клікає на посилання. Якщо цей метод не працює, потрібно створити мережу серверів з різними IP-адресами і робити запити у випадковому порядку з різних серверів. В такому випадку, сервер-відповідач не зможе визначити, що це один й той самий користувач.

По-третє, багато ресурсів потребують, щоб запит до них був відмічений, як браузерний. Оскільки під час запитів сервер-сервер за замовчуванням немає додаткової інформації, необхідно додати заголовок User-Agent, у значення якого потрібно вставити будь який агент браузера.

Наприклад при реалізації парсеру для нашого прикладу для аналізу даних із «ASOS» стало необхідним додавання заголовку Cookies. В ньому зберігається кука crumb, яка підписує кожен запит до серверу. Без такого підпису, сервер відмовляє у відповіді.

4.2 Вибір технологій

Мовою програмування для написання агрегатору для пошуку товарів, із використанням кластеризації даних об’єктів за зображенням із урахуванням найкращої ціни та моди, було обрано JavaScript, для побудови клієнтської частини було обрано бібліотеку React.js, а для реалізації серверної частини – фреймворк Node.js.

JavaScript – інтерпретована об'єктно-орієнтована мова програмування з динамічною типізацією. Дану мову програмування починають усе частіше використовувати в області штучного інтелекту і з її використанням створюються прогресивні та сучасні бібліотеки та фреймворки для роботи із нейронними алгоритмами, алгоритмами машинного навчання, алгоритмами кластеризації та, взагалі, для проведення різного роду статистичних обчислень.

Інтерпретована мова програмування – мова програмування, в якій похідний код програми не перетворюється попередньо повністю у машинний код для виконання, як у компільованих мовах, а виконується рядок за рядком за допомогою спеціальної програми-інтерпретатора. Головні відмінності між компільованими й інтерпретованими мовами:

– швидкість виконання програми, компільованої в машинний код, перевершує швидкість інтерпретованої програми, як правило, в десятки і сотні разів;
– у разі використання компілятора, при внесенні змін у похідний код програми, перш ніж ці зміни можна буде побачити в роботі програми, необхідно виконати компіляцію похідного коду.

У загальному випадку, будь-яка мова може бути компільованою і інтерпретованою, так що це розділення належить до практики застосування мови, а не є його властивістю.

Серед основних переваг даної мови програмування є наступні:

– чистий, C-подібний синтаксис;

– має кілька менеджер пакетів, таких, як npx, npm та yarn, які мають велику кількість корисних модулів;

– має велике комьюніті.

JavaScript має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування, у основі якого лежить прототипне наслідування. Елегантний синтаксис JavaScript, динамічна обробка типів, а також те, що це нтерпретована мова, роблять її ідеальною для написання скриптів та швидкої розробки прикладних програм у багатьох галузях на більшості платформ, зокрема, розробки та прототипуванні програм у галузі кластерного аналізу даних.

Node.js – фреймфорк, заснована на рушію V8 (здійснює трансляцію JavaScript в машинний код), що перетворює JavaScript з вузькоспеціалізованої мови в мову загального призначення. Node.js додає можливість JavaScript взаємодіяти з пристроями введення-виведення через свій API (написаний на C ++), підключати

інші зовнішні бібліотеки, написані на різних мовах, забезпечуючи виклики до них з JavaScript-коду. Node.js застосовується переважно на сервері, виконуючи роль веб-сервера, але є можливість розробляти на Node.js і комп'ютерні віконні додатки (за допомогою NW.js, AppJS або Electron для Linux, Windows і macOS) і навіть програмувати мікроконтролери (наприклад, tessel і espruino).

React.js – це бібліотека від команди розробників Facebook, написана на мові програмування JavaScript, для побудови інтерфейсу клієнта. Дана бібліотека є open-source проектом, тобто – це проект з відкритим похідним кодом, тобто кожен може безкоштовно завантажити та змінити його вихідний код, виправити помилки та запропонувати свої зміни, створивши пул-реквест. Розмір бібліотеки React.js може зростати експоненціально за допомогою доповнених файлами бібліотек, підготовлених спільнотою React.js, починаючи від колекцій індивідуальних функцій інтерфейсу і закінчуючи шаблонами React.js для створення шаблонів інтерфейсу з нуля.

Основною перевагою використання React.js для побудови клієнтського інтерфейсу є те, що дана бібліотека використовує Virtual DOM.

Virtual DOM – це концепція програмування, в якій ідеально або «віртуальне» представлення користувальницького інтерфейсу зберігається в пам'яті та синхронізується з «справжнім» DOM при допомозі бібліотеки, такому як ReactDOM. Такий підхід і робить API React.js декларативним: ви вказуєте, у якому стані має знаходитись користувальницький інтерфейс, а React.js отримує, щоб DOM відповідав цьому рівню. Це абстрагує маніпуляцію з атрибутами, обробляє вміст і ручне оновлення DOM. Завдяки використанню цієї концепції, запобігається зайвий рендеринг DOM дерева, що прискорює роботу застосунку, бо операції з DOM дуже важкі, та шкодять продуктивності.

React.js спрощує загальний процес створення сценаріїв компонентів. Сценарій клієнтської частини застосунку набагато зручніший і кращий, коли використовується розширення синтаксису під назвою JSX. Цей синтаксис робить рендеринг компонента простішим завданням. Завдяки набору шаблонів для написання програми. Він є синтаксично більш зрозумілим, бо дає змогу

використовувати HTML нотацію у документі з JavaScript кодом. Та покращує процес перевикористання компонентів. Це полегшує технічне обслуговування та підвищує продуктивність. Найчастіше оновлення та підтримка перетворюються на складний процес, оскільки додаток має складну логіку, і будь-яка модифікація одного компонента може вплинути на інші.

React.js гарантує більш швидку візуалізацію, по підтримує асинхронний рендеринг, тому користувачеві не потрібно дожидатись, повного інтерпретування документа, бо він має змогу бачити результат на сторінці за необхыдностю, тобто спочатку завантажуються дані, що знаходяться у першій області, яку може побачити користувач, а потім уже усі останні компоненти.

Підтверджено, що бібліотека React.js забезпечує кращий досвід користувача, більш високу продуктивність програми та затрачується менше часу на написання коду.

4.3 Вибір бази даних

База даних – це сукупність матеріалів, представлених в об'єктивній формі і систематизованих таким чином, щоб ці матеріали можна було знайти і обробити.

Бази даних можна класифікувати за такими параметрами:

– модель даних;

– ступінь розподіленності;

– вміст.

Якщо брати до уваги параметр за моделлю даних докладніше, то можна поділити на такі групи:

– реляційна база даних;

– документо-орієнтована база даних;

– ієрархічна база даних тощо.

Більш детально розглянемо два типи БД – реляційну і документо-орієнтовану.

Реляційна база даних – це база даних, заснована на реляційній моделі. Принцип полягає в зберіганні даних у вигляді таблиць з зумовленими зв'язками між ними. Кожен рядок в таблиці є набором зв'язаних значень, що відносяться до однієї сутності. З переваг можна виділити SQL – мова запитів, яка дозволяє досить гнучко маніпулювати даними. Незаперечною перевагою є цілісність і повнота даних, які досягаються за допомогою використання ключів, а також правил нормалізації, які потребують приведення інформації, що зберігається до чітко структурованого формату.

Документо-орієнтована база, на відміну від реляційної, будується на основі документів. Документами можуть виступати будь-які ієрархічні структури. Основою таких баз даних є документальне сховище, яке організоване у вигляді дерева. Документи як правило групуються в колекції, аналоги таблиць в реляційних базах. Однак ключовою відмінністю є те, що в колекцію можна записати документи абсолютно будь-якого формату і змісту. Перевагою таких баз даних є простота використання, висока швидкість роботи, дуже гнучку мову запитів, яка дозволяє об'єднувати і групувати документи в залежності від завдання, а також можливість звернення до бази даних і пошуку по документам без повного вивантаження документів в оперативну пам'ять.

В якості основного сховища даних в рамках дипломної роботи буде використовуватися документо-орієнтована база даних MongoDB. Вибір на користь цього рішення зроблений, тому що це дозволить найбільш просто і швидко реалізувати необхідний функціонал без втрати продуктивності.

MongoDB – документо-орієнтована база даних, яка забезпечує високу продуктивність і легку масштабованість. В основі даної БД лежить концепція колекцій і документів. База даних представлена у вигляді фізичного сховища колекцій. Кожна БД має свій власний набір файлів в файлову систему. Зазвичай, один MongoDB сервер має кілька БД.

Колекція – це група документів MongoDB. Є еквівалентом простої таблиці в реляційної базі даних. Колекція міститься всередині однієї БД. Документ в колекції може мати різні поля. Найчастіше, всі документи в колекції створені для однієї цілі.

Документ – це набір пар ключ – значення. Документ має динамічну схему. Це означає, що документ в одній і тій же колекції не зобов'язаний мати однаковий набір полів або структуру, а загальні поля в колекції можуть мати різні типи даних.

Приклад документу, який був створений після парсингу сайту ASOS та збережений до бази даних, можна побачити на рисунку 4.4.

На рисунку 4.4 можна побачити структуру документу.

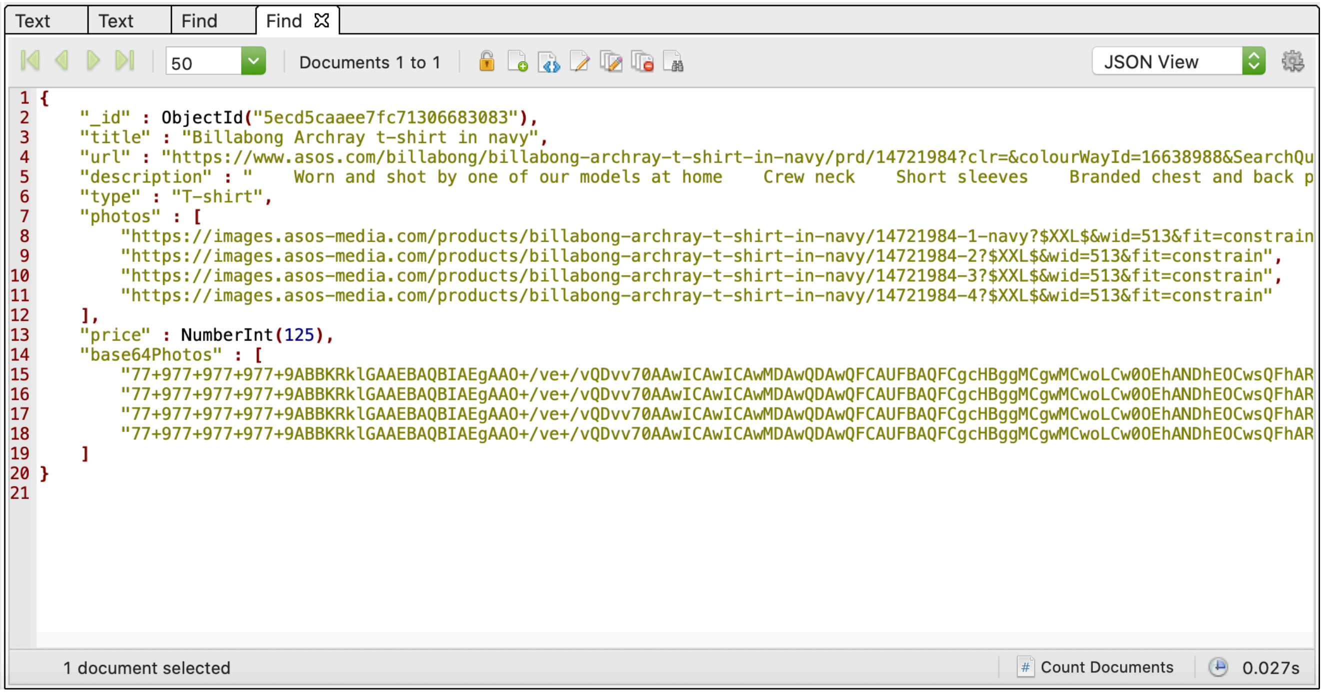

Рисунок 4.4 – Документ, що містить дані про продукт

Він потрібен для обробки даних та буде використовуватись для пошуку.

4.4 Реалізація алгоритму кластеризації

Алгоритм схематично зображено на рисунку 4.5.

Можна побачити, що після надходження зображення, першим кроком є попередня обробка зображення.

Попередня обробка зображення – це початковий крок нашого алгоритму, коли кольорові зображення піддаються шуму та будь-яким іншим нерівностям, затушовуючи потрібну інформацію. Під час формування зображень вони можуть стикатися із неточностями через низьку освітленість, розмиття та механічний шум. Вхідні зображення попередньо обробляються, видаляючи такі помилки, згладжуючи ефект шуму з подальшим посиленням краю для кращих результатів на пізніших етапах.

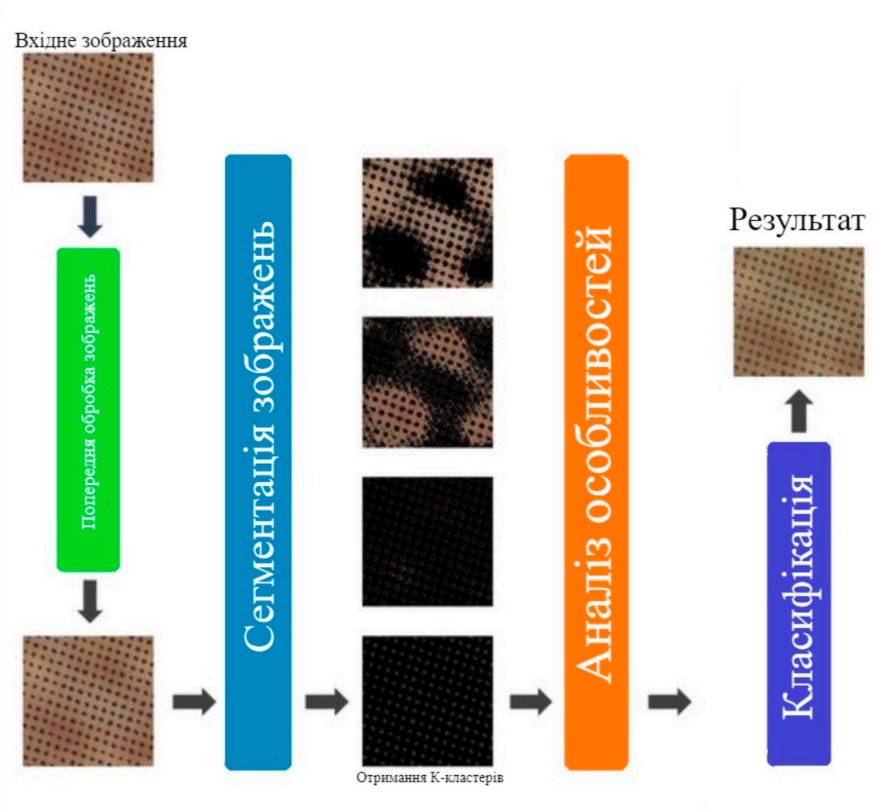

Рисунок 4.5 – Зображення кроків алгоритму

Для попередньої обробки зображення було обрано провести покращення зображень за допомогою Гауссівського Лапласіана (Laplacian of Gaussian), або LoG.

Лапласіанський фільтр використовується для виділення областей, що демонструють різкі зміни рівня інтенсивності, що призводить до посилення країв

зображення [17]. Враховуючи чутливість Лапласіанський фільтра до шуму, фільтр Гаусса використовується як оператор згладжування для нормалізації шуму в межах зображення. 2D фільтр Гаусса наведено у формулі (4.1).

$$G(x,y) = \frac{1}{2\pi\sigma^2}\, e^{\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)}, \tag{4.1}$$

де, σ – стандартне відхилення.

Результат згортки фільтра Гаусса із вхідним зображенням im(x, y) наведено у формулі (4.2).

$$L(x,y) = im(x,y) * G(x,y), \tag{4.2}$$

де, L(x, y) – Гауссове представлення простору в масштабі вхідного зображення im(x, y);

* – являє собою оператор згортки.

Гауссове згладжування з наступним використанням фільтру Лапласіана можна поєднувати за допомогою одного оператора, відомого як Гауссівського Лапласіана (LoG), що показано в рівнянні (4.3).

$$\nabla^2\, G(x,y) = \frac{x^2+y^2-2\sigma^2}{\pi\sigma^4}\, e^{\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)}, \tag{4.3}$$

де, x і y – просторові координати зображення;

σ – являє собою стандартне відхилення.

Застосування фільтра Гаусса перед Лапласіаном зменшує шум, тим самим покращуючи продуктивність Лапласіанового оператора посилення краю [17].

Другим кроком є сегментація зображення. У нашому випадку використовується метод сегментації за допомогою кластеризації k-середніх.

Метод кластеризації K-сердніх ділить дані на групи, що не перетинаються. Алгоритм виконує ітераційне присвоєння кожної точки даних одній із груп K на

основі найменшої відстані центроїдів до їх простору функцій. Процес повторюється до тих пір, поки центроїди не досягнуть своїх кінцевих постійних положень. Для введення K кількості кластерів та відповідних центроїдів точки даних, що представляють товари, що маємо для пошуку, кластеризуються та витягуються. Схематично представлено блок-схему алгоритму кластеризації на рисунку 4.6.

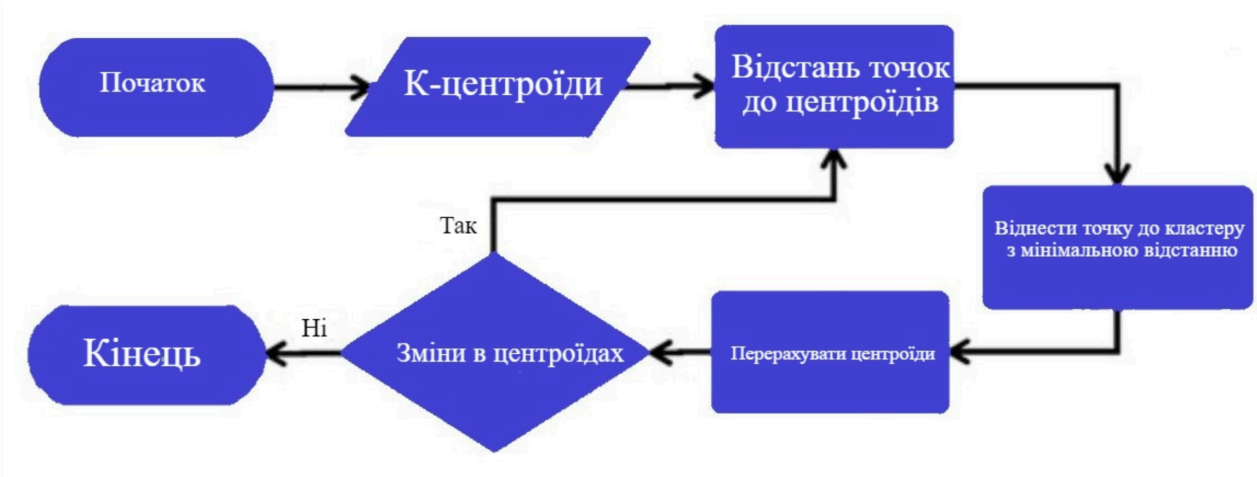

Рисунок 4.6 – Блок-схема алгоритму

Центроїд – для кластеру, це ключова точка даних, навколо якої здійснюється кластеризація. Він схожий з будь-якою іншою точкою даних, представленою векторною ознакою, обраної випадковим чином або через певних механізм.

Існують різні методи визначення відстані до найближчого центроїду. Один з найбільш використовуваних методів є знаходження Евклідової відстані. Після групування проводиться перерахунок нового центроїду для кожного кластеру, а нова евклідова відстань обчислюється між кожним центром і кожної точки даних і призначає точки кластеру, які мають мінімальну евклідову відстань.

Тому k-середніх – це ітераційний алгоритм, в якому він мінімізує значення суми відстаней від кожного об'єкта до центру кластера [18].

Розглянемо реалізацію для сегментації зображення.

Розглянемо зображення розміром x на y. Дане зображення повинно бути кластеризовано на k кількість кластерів. Нехай, $p^i(x, y)$ – вхідні пікселі для

кластеризації для кожного із каналів RGB, для R, G, B , де *i=3,* та C_k^i – центр кластеру *i* каналу кольорового зображення. Наступними кроками обробки зображення є:

а) поділити зображення на 3 канали Red(червоний), Green(зелений), Blue(блакитний);

б) проініціалізувати кількість кластерів *k* та центр для кожного каналу;

в) для кожного пікселю зображення потрібно вирахувати Евклідову відстань *d* між центром та кожним пікселем, використовуючи формулу 4.4 наведену нижче:

$$d_k = \sum_{i=1}^{3} \left\| p^i(x,y) - C_k^i \right\| \tag{4.4}$$

г) Віднести пікселі до найближчого з центроїдів;

д) після того, як всі пікселі будуть віднесені до центрів, вирахувати нові центроїди шляхом усереднення пікселів. Формула усереднення (4.5) наведена нижче:

$$C_k^i = \frac{1}{k} \sum_{y \in C_k^i} \sum_{x \in C_k^i} p^i(x,y) \tag{4.5}$$

е) повторити процес.

Фрагменти класу реалізації кластеризації наведено нижче:

```
export class Kmeans {
constructor(points, clustersCount=3){
     this.data = clone(points);
this.length = this.data.length;
this.clusters = this.initClusters();


  }

step() {
    for (let i = 0; i < length; ++i) {
this.addToNearestCluster(data[i]);
    }
```

```
        const changedClusters = recomputeCentroids();
        const hasMoreWork = changedClusters > 0;

        return hasMoreWork;
      }

    recomputeCentroids() {
        let changedClusters = 0;
        for (let i = 0; i <  clustersCount; ++i) {
          changedClusters
    +=  this.clusters[i].recomputeCentroid();
        }
        return changedClusters;
      }

      addToNearestCluster(pt) {
        let min = Number.POSITIVE_INFINITY;
        let idx = -1;

        for (let i = 0; i < clustersCount; ++i) {
          const                          dist                          =
this.clusters[i].getEucDistance(pt.cluster[i],pt.center.center);
          if (dist < min) {
            min = dist;
            idx = i;
          }
        }

        if (idx !== pt.cluster) {
          if (pt.cluster !== undefined) {
    this.clusters[pt.cluster].remove(pt);
          }
    this.clusters[idx].add(pt);
    pt.cluster = idx;
        }
      }

    getClusters() {
        return this.clusters;
      }
```

Після того, як отримали сегменти, можна розглянути особливі ознаки, завдяки яким і буде виконуватись порівняння для пошуку товарів агрегатору.

Для кожного сегменту(кластеру) можна виділити такі особливі ознаки, які будуть корисні для подальшої обробки:

– середнє по червоному(R) каналу – середнє значення пікселів, що належать червоному каналу;
– середнє по блакитному(B) – середнє значення пікселів, що належать блакитному каналу;

- середнє по зеленому(G) каналу – середнє значення пікселів, що належать зеленому каналу;
- площа – кількість пікселів у сегменті(кластері);
- комбіноване значення кольору – середнє значення кольору сегменту;
- Координати реберних точок – масив граничних точок.

Завдяки отриманим особливостям, у момент, коли користувач обирає пошук за зображенням, зображення отримане із запиту користувача, також проходить процес сегментації, та на основі отриманих сегментів(кластерів), відбувається пошук серед існуючих айтемів у базі даних. Виконується порівняння усіх факторів особливостей та на їх базі формується масив результату пошуку, тобто схожих товарів, але важливо зауважити, що значення площі повинно бути відносним та ураховуватися в останню чергу, бо залежно від ракурсу та типу фотографії вони можуть різнитись, тому важливо мати інформацію про ключові точки, бо їх можна зіставити для порівняння.

Коли формується масив товарів, які можуть задовольняти пошуковий запит за зображенням, потрібно відсортувати товари за найкращою ціною. Але що означає поняття найкращої ціни, для даного випадку значення має на увазі, що ціна є найнижчою із тих, що було отримано після парсингу.

Також потрібно мати можливість відсортувати результати за модою.

Під модою мається на увазі найбільш популярні айтеми, популярність визначається за кількістю відгуків на товар, або якщо при парсингу товар був у категорії найбільш популярних товарів.

4.5 Створення інтерфейсу користувача

Для реалізації інтерфейсу клієнтської частини було вирішено реалізувати максимально мінімалістичний та зрозумілий дизайн.

При попаданні на сторінку агрегатору, зображення інтерфейсу можна побачити на рисунку 4.7 , перше що бачить користувач – список айтемів, які є у базі, поле пошуку за назвою товару, 2 простих фільтри, що сортують значення за популярністю та ціною. При наведенні на об’єкт з’являється напівпрозорий блок, у якому міститься інформація про повну назву товару та її ціну.

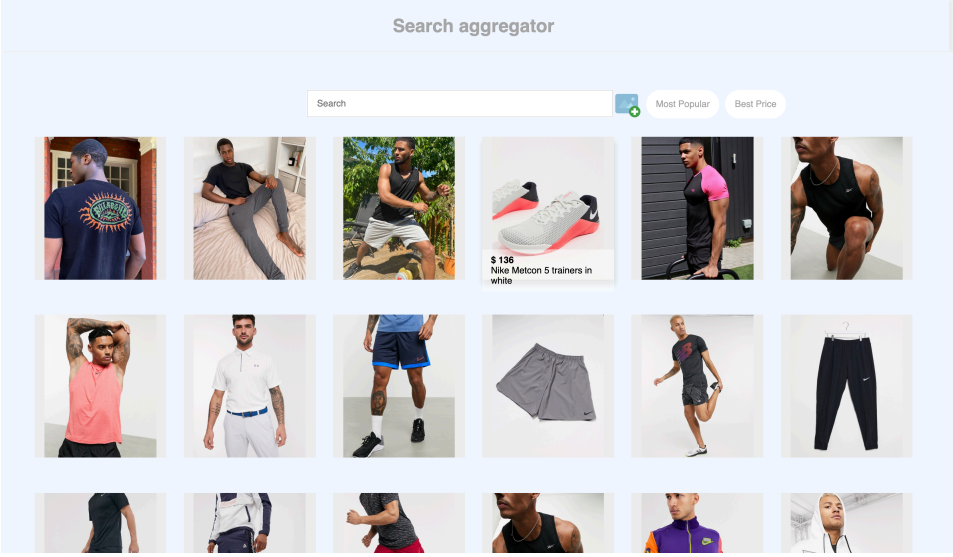

Рисунок 4.7 – Початковий вид сторінки

Також, біля поля для текстового пошуку можна побачити іконку із зображенням, натиснувши на яку з’явиться поле для вибору фотографії для пошуку. Існує можливість або завантажити зображення, відкривши пошуковий діалог, або використовуючи технологію Drag and Drop перетягнути зображення до області, спеціально визначеної. Формати, які підтримуються для аналізу – це jpeg, png. Вигляд сторінки з можливістю для пошуку за зображенням можна побачити на рисунку 4.8.

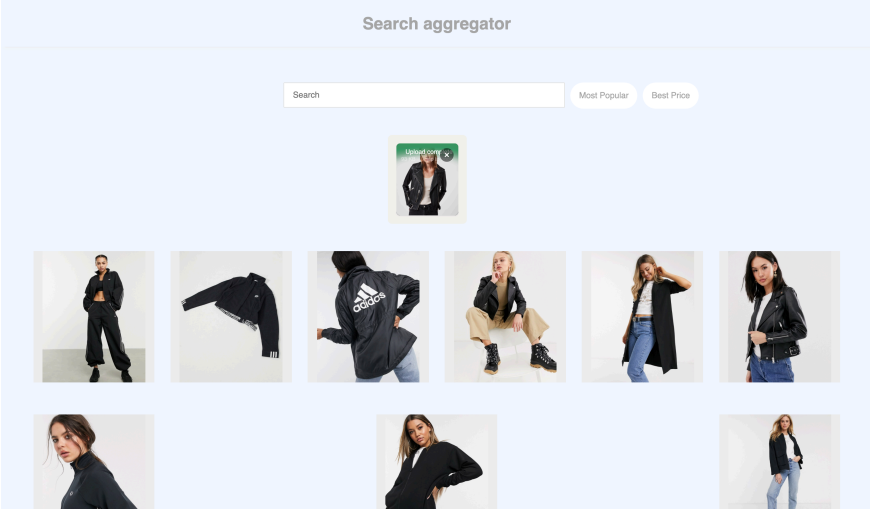

Рисунок 4.8 – Інтерфейс з пошуком за зображенням

Щоб отримати детальну інформацію про об’єкт, по натисканню на нього, з’являється модальне вікно (рисунок 4.9).

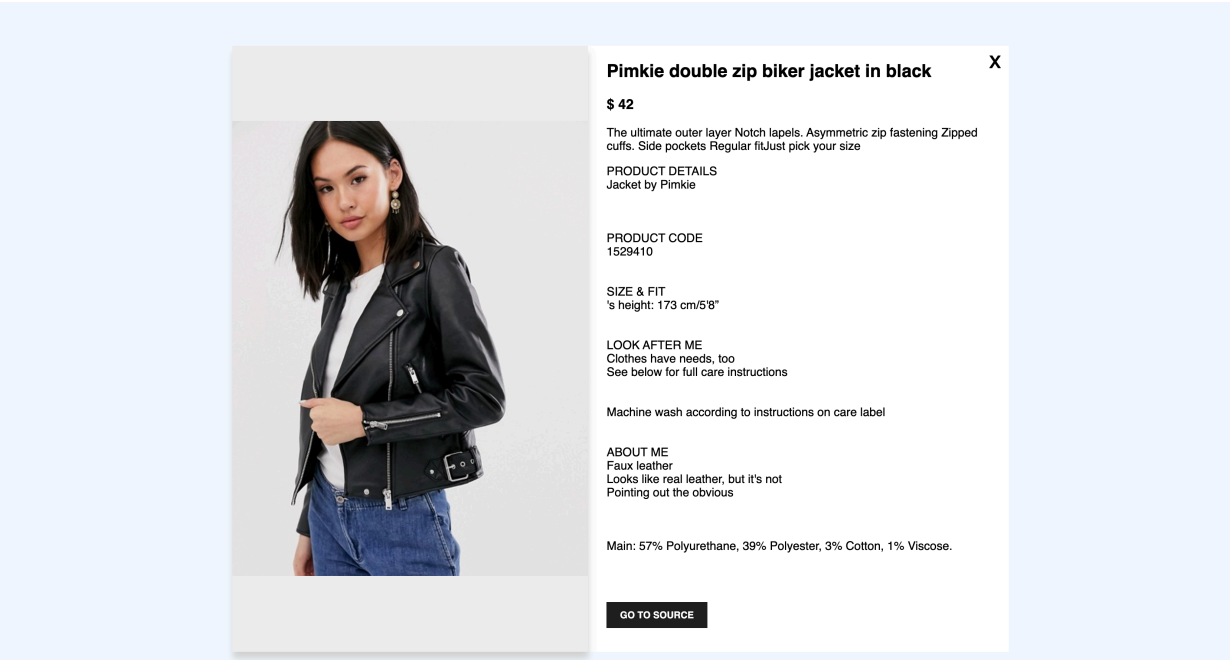

Рисунок 4.9 – Модальне вікно з товаром

В цьому вікні міститься інформація про товар, його назва, ціна, опис, та посилання на ресурс, з якого було представлено товар.

ВИСНОВКИ

Метою роботи була реалізація агрегатору для пошуку товарів серед даних, що були отримані із різних ресурсів, з підтримкою можливості пошуку за зображенням із використанням кластеризації даних та підтримкою урахування найкращої ціни та моди.

У ході дипломної роботи було створено модель програмного забезпечення, яке надає змогу знайти товар за зображенням. Були проаналізовані існуючі аналоги, які вже існують на ринку, їх переваги та недоліки.

Для отримання даних із різних ресурсів було створено парсер, який зчитує дані із DOM та аналізує отриманий текст, витягує потрібну інформацію та перетворює її у документ(об’єкт) для зберігання у базі даних MongoDB.

Для реалізації функції пошуку за зображенням, було використано алгоритм сегментації зображень, який базується на алгоритмі кластеризації даних k-середніх.

При реалізації сегментації із використанням алгоритму кластеризації k-середніх, був реалізований підхід з розподіленням на кластери, спираючись на кольорові особливості зображення та розподілення на 3 канали RGB спектру, тобто R – червоний, G – зелений, B – блакитний.

Перевагою використання алгоритму кластеризації k-середніх є те, що він простий і більш швидкий для обчислювання, ніж, наприклад, ієрархічний підхід. Також він може працювати для великої кількості змінних. Але може давати різні результати. Тому потрібно ініціалізувати належну кількість кластерів. Різне значення початкового центроїду призведе до отримання різних кластерів. Тож вибір належного початкового центроїду є дуже важливим завданням.

На даний момент немає універсального методу сегментації кольорових зображень. Бо більшість вхідних даних зображення складаються з багатьох областей. А деякі образи чітко не видно, або на деяких з них присутній шум або мають погану якість. Тому вони потребують попередньої обробки перед тим, як починати сегментація.

К-середніх має як переваги, так і деякі недоліки. Якість результату кластеризації залежать від початкового вибору початкових центроїда. Тож, якщо початковий центроїд буде обраний випадковим чином, він буде отриман різний результат для різних початкових центрів.

У процесі виконання дипломної магістерської роботи, було знайдено такий недолік, як неможливість розпізнати потрібну текстуру для пошуку товару, а також те, що на деяких зображеннях, де зображення товару представлено під іншим кутом, ніж на зображенні, яке було отримано для пошуку схожих, алгоритм працює з великою похибкою. Але у цілому, зазначені негативні моменти не стають перешкодою для використання додатку.

Можливими варіантами подальшого розвитку даної роботи можуть бути використання білінійної або сиамської згорткової нейронної мережі. Можливо у подальшому спробувати використання композицій та ансамблів нейронних мереж та класичних алгоритмів машинного навчання. Однак, для цього буде потрібно набагато більше високопродуктивних обчислень та ресурсів, аніж ті, що було використано в даній роботі [19].

У процесі виконання роботи, було створено простий та інтуїтивно зрозумілий інтерфейс. Користувач, потрапивши на сторінку застосунку одразу бачить усі об’єкти, що були отримані за допомогою парсингу зовнішніх сайтів, у верхній частині сторінки користувач може використати пошук за текстом, пошук буде проводитись по назві товару, біля текстового поля для пошуку, натиснувши на зображення фотографії, користувач побачить секцію для можливості завантаження зображення для пошуку. Правіше від іконки для пошуку за зображенням існує 2 фільтри для сортування результату пошуку за найкращою ціною та модою.

Тематика даної роботи є актуальною, продовження досліджень в цій сфері має бути перевірка нових способів класифікації даних та їх аналізу.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Юрасов А.В. Основы электронной коммерции. – М.: Горячая линия – Телеком, 2008. – 480 с.

2.Форсайт Д.А., Понс Дж. Компьтреное зрение. Современный подход. Пособие – Вильямс, 2018. – 960 с.

3. The 5 Clustering Algorithms Data Scientists Need to Know. – оновлення: 7.11.2018. URL: https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68

4. Чубукова И.А. Data Mining: учеб. пособие. М.: ИНТУИТ. БИНОМ. Лаборатория знаний, 2006. – 382с.

5. Huiz Zhu, Francis H.Y. Chan and F.K.Lam, «Image Contrast Enhancement by Constrained Local Histogram Equalizatio» in Computer Vision and Image Understanding, Vol. 73, No. 2, Febbruary. 1999. – 290c.

6. Князь, Димитрия Анализ основных алгоритмов кластеризации многомерных данных / Димитрия Князь. – М.: LAP Lambert Academic Publishing, 2014. – 795 с.

7. Sai Krishna, T.V.; Yesu Babu, A.; Kiran Kumar, R. Determination of Optimal Clusters for a Non-hierarchical Clustering Paradigm K-Means Algorithm. In Proceedings of International Conference on Computational Intelligence and Data Engineering; Lecture Notes on Data Engineering and Communications Technologies; Chaki, N., Cortesi, A., Devarakonda, N., Eds.; Springer: Singapore, 2018 – 417p.

8. Bhosale, N.P.; Manza, R.R. A Review on Noise Removal Techniques From Remote Sensing Images. In Proceedings of the Radhai National Conference [CMS], Aurangabad, India, 2013. – 12p.

9. Детекторы и дескрипторы особых точек FAST, BRIEF, ORB. – оновлення: 18.06.2018. URL: https://habr.com/ru/post/414459/

10. Sherbrooke, D. RANKED K-MEANS CLUSTERING FOR TERAHERTZ IMAGE SEGMENTATION Mohamed Walid, Canada. International Conference on Image Processing (ICIP), 2015. – 4395p.

11. Li, Y., & Shen, Y. Fuzzy c-means clustering based on spatial neighborhood information for image segmentation. Journal of Systems Engineering and Electronics, 21(2), 2010. – 323–328p.

12. Li, B., Zhang, Y., Lin, Z., & Lu, H. Subspace clustering by Mixture of Gaussian Regression. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2015. – 2094–2102p.

13. Deep Claster – алгоритм глубокой кластеризации. Он почти достиг человеческой точности. – оновлення: 19.10.2018. URL: https://neurohive.io/ru/papers/deep-claster/

14. Алгоритм быстрого нахождения похожих изображенийАлгоритм быстрого нахождения похожих изображений. – оновлення 22.06.2011 – URL: https://habr.com/ru/post/122372/

15. Introduction to Image Segmentation with K-Means clustering. – оновлення 29.07.2019. URL: https://towardsdatascience.com/introduction-to-image-segmentation-with-k-means-clustering-83fd0a9e2fc3

16. Wang J. Z., Li J., Wiederhold G., «SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture Libraries», IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 9, Sept. 2001. – 947-963 p.

17. Veenman C. J., Reinders M. J. T., Backer E., “A Maximum Variance Cluster Algorithm”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 9, Sept. 2002. – 1273-1280 p.

18. Jain A., Murty M., Flynn P. Data Clustering: A Review. // ACM Computing Surveys. 1999. – 31p.

19. Дж. Ту, Р. Гонсалес «Принципы распознавания образов», Издательство «Мир», Москва 1978. 109 – 112 р.

20. Ревенчук І.А., Перцьова К.В., Маренич О.І. Програмна реалізація кластеризації пошукових запитів.-Біоніка інтеллекту.-№.-2(91)2018.-С.86-93.

21. Білоконенко В.М , Ревенчук І.А. Алгоритми сегментації зображень на базі побудови матриць збігів. Східно-Європейський журнал передових технологій.-№2(62), том 2.-2013. – С.43–45.

22. Ревенчук І.А. Математична модель агрегації даних в соціальних медіа//Міжнарод. Наук. Конф. ISCOPT-2017 “Питання оптимізації обчислень (ПОО-XLIV)”. – Кам'янець-Подільський. – 2017. С.197-203.

23. Bohdan Sus, Nataliia Tmienova, Ilona Revenchuk, Vira Vialkova. Development of Virtual Laboratory Works for Technical and Computer Sciences //ICIST - International Conference on Information and Software Technologies. – Vilnius, 10-12 October, 2019. – P. 383-394.