Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

РОЗРОБКА ТА ДОСЛІДЖЕННЯ МЕТОДУ ВИЯВЛЕННЯ ПІДОЗРІЛИХ НА ПЛАГІАТ ЗОБРАЖЕНЬ В ЕЛЕКТРОННИХ ДОКУМЕНТАХ
(тема)

Виконав:
студент 2 курсу, групи ІНФм-21-1

Попирєв Д. О.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Яковлева О. В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Попирєву Даниїлу Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка та дослідження методу виявлення підозрілих на плагіат зображень в електронних документах .
затверджена наказом по університету від 9 листопада 2022 року № 1469Ст
2. Термін подання студентом роботи до екзаменаційної комісії 21 листопада 2022 р.
3. Вихідні дані до роботи математичні моделі отримання дескрипторів: SIFT, AKAZE, методи пошуку відповідних дескрипторів: *k*-найближчих та NNDR, метод усунення хибних відповідностей: RANSAC, навчена нейромережа для класифікації зображень: MobileNetV2, Java, Python, React, PostgreSQL, OpenCV, Keras.

4. Перелік питань, що потрібно опрацювати в роботі .
1. Вивчити питання кластеризації зображень.
2. Дослідити питання поділу зображень на класи на основі нейромережевого підходу.
3. Ознайомитися із використання навченних заздалегідь нейронних мереж.
4. Дослідити використання гістограм у якості ознак зображень для подальшого етапу кластеризації.
5. Розробити поетапний класифікатор для аналізу вхідних зображень.
6. Сформувати датасет для перенавчання моделі нейронної мережі.
7. Спроєктувати та розробити застосунок для пошуку плагіату зображень в електронних документах.
8. Сформувати базу даних зображень для дослідження роботи застосунку.
9. Провести дослідження швидкості та якості роботи застосунку.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) опис датасетів зображень, приклади виділення дескрипторів та побудови відповідностей, оцінки швидкодії.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Консультант з дотримання діючих стандартів та норм	Доцент Творошенко І.С.		

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	09.11.2022	
2	Аналіз завдання, підбір літератури	09.11.21-11.11.21	
3	Аналіз літератури з досліджуваної проблеми	11.11.21-14.11.21	
4	Аналіз технічних засобів	14.11.21-15.11.21	
5	Розробка методу	15.11.21-17.11.21	
6	Програмна реалізація	17.11.21-20.11.21	
7	Оформлення пояснювальної записки	20.11.21-23.11.21	
8	Перевірка на плагіат	24.11.2021	
9	Рецензування	25.11.2021	
10	Підготовка презентації та доповіді	26.11.2021	
11	Занесення роботи в електронний архів	27.11.2021	
12	Попередній захист кваліфікаційної роботи	30.11.2021	

Дата видачі завдання 9 листопада 2022 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Яковлева О. В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 70 с., 34 рис., 1 дод., 42 джерела.

ДЕТЕКТОРИ, ДЕСКРИПТОРИ ПОВНОГО ЦИКЛУ, SIFT, AKAZE, МЕТОД *K*-НАЙБЛИЖЧИХ СУСІДІВ, NNDR, RANSAC, ПЛАГІАТ, КЛАСИФІКАЦІЯ, НЕЙРОННІ МЕРЕЖІ OPENCV, MOBILENETV2, PYTHON.

Об’єктом дослідження є питання кластеризації зображень, для найбільш точного їх віднесення до конкретної групи.

Метою дослідження є розробка та дослідження методу виявлення підозрілих на плагіат зображень в електронних документах.

У ході попередньої роботи було створено прототип застосунку, який знаходив підозрілі на плагіат зображення, використовуючи дескриптори SIFT та AKAZE. Проте у прототипу була одна вада – швидкодія.

Для вирішення цієї проблеми, було змінено алгоритми попередньої обробки зображень, для більш точної локалізації можливих збігів із зображеннями у базі даних.

У результаті проведеної роботи було покращено розроблений раніше прототип програмного застосунку для пошуку плагіату зображень у електронних документах.

DETECTORS, FULL CYCLE DESCRIPTORS, SIFT, AKAZE, *K*-NEAREST NEIGHBORS, NNDR, RANSAC, PLAGIARISM, CLASSIFICATION, NEURAL NETWORKS OPENCV, MOBILENETV2, PYTHON.

The object of the research is the issue of image clustering, for the most accurate assignment of them to a specific group.

The aim of the research is to develop and research a method for detecting suspected plagiarism images in electronic documents.

As a result of the previous work, an application prototype was created that found suspected plagiarized images using SIFT and AKAZE descriptors. However, the prototype had one flaw – speed.

To solve this problem, the image pre-processing algorithms were changed, for more accurate localization of possible matches with images in the database.

As a result of the work carried out, the previously developed prototype of a software application for searching for plagiarism of images in text documents was improved.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

SUSAN – Smallest Univalue Segment Assimilating Nucles (асимілююче ядро найменшого однозначного сегмента)

FAST – Features From Accelerated Segment Test (функції прискореного сегментного тесту)

SIFT – Scale-Invariant Feature Transform (масштабно-інваріантне перетворення ознак)

SURF – Speeded Up Robust Features (прискорені надійні функції)

BRIEF – Binary Robust Independent Elementary Feature (двійкова надійна незалежна елементарна функція)

ORB – Oriented FAST and Rotated BRIEF

БД – база даних

СУБД – система управління базами даних

ШНМ – штучна нейронна мережа

ЗНМ – згорткова нейронна мережа

ВСТУП

Унікальність думок, їх цінність та важливість це – саме те, про що слід піклуватися в наш час – еру інформаційних технологій. З метою їх збереження, створюються патенти, а текстові видання, такі як наукові роботи чи журналістські статті, навіть рекламу, перевіряють на плагіат [1]. Проте найчастіше, коли ми говоримо про друковані роботи, у цих перевірках розглядається лише текстова частина документу. Вже існують сервіси для пошуку зображень за зображенням-зразком, такі як Google Search by Image, TinEye, Yandex Image. Але ці сервіси не вміють працювати із зображеннями, що містяться в текстових документах, зокрема наукових роботах, що можуть містити як звичайні зображення, так і, що є найчастіше, схематичні.

На цей час залишається невирішеною задача пошуку плагіату зображень у документах, коли на вхід подаються електронні документи, наприклад, формату .doc або .pdf, що містять зображення, та у якості результату отримується інформація про наявність зображень, що є підозрілими на плагіат.

Робота присвячена вирішенню проблеми пошуку у електронних документах підозрілих на плагіат зображень. Ця проблема відноситься до задач пошуку зображень на основі їх змісту без використання текстового опису.

В роботі пропонується для вирішення задачі використовувати дескрипторний підхід, оскільки дескриптори у сумісному застосування з методами NNDR та RANSAC дозволяють визначати схожість зображень з високою точністю.

Попередньо був проведений порівняльний аналіз роботи дескрипторів, наявних у відкритій бібліотеці OpenCV – SIFT, SURF, BRISK, ORB, KAZE, AKAZE, у контексті даної проблеми. При дослідженні порівнювалися їх точність та швидкість роботи, повторюваність виявлення ключових точок, схожість їх опису, питання щодо виявлення відповідних пар дескрипторів,

відхилення хибних відповідностей, стійкість до геометричних змін зображення, серед яких: масштабування, обрізання частин зображення, зміну формату зображення, у тому числі при збереженні його у документі. У результаті дослідження, було обрано дескриптори SIFT та AKAZE, як найбільш підходящі для вирішення задачі пошуку зображень, підозрілих на плагіат. Їх відмінність полягає у тому, що AKAZE є менш точним, але швидшим у порівнянні з SIFT.

Метою дослідження є розробка та дослідження методу виявлення підозрілих на плагіат зображень в електронних документах. Особливу увагу необхідно приділити швидкості пошуку зображень за рахунок попередньої кластеризації зображень в електронній колекції [2].

1 ОГЛЯД ПРОБЛЕМАТИКИ ПОСТАВЛЕНОЇ ЗАДАЧІ

1.1 Задача пошуку зображень, успіхи в її вирішенні та проблеми пошуку плагіату зображень у текстових документах

Епоха інформаційних технологій – це час, коли з’являється все більше і більше даних, їх оцифровують для легшого зберігання, обробки та подальших маніпуляцій ними. Й звісно, завдяки інтернету, ці дані легко і швидко поширюються серед населення всієї планети. До таких даних відносяться й різні відео-матеріали, зокрема зображення. З ними найчастіше виконують наступні операції:

- пошук інформації, пов’язаної із зображенням, в інтернеті;
- аналіз зображень (у медичних чи військових цілях);
- пошук зображень в інтернеті;
- геолокація;
- організація каталогів та маніпулювання ними;
- відстеження зображень з авторськими правами й їх захист.

Для виконання цих задач було розроблено велику кількість різних алгоритмів та методів пошуку та аналізу. Деякі з них використовують як основу ознаки зображень. Це можуть бути дескриптори характерних точок, яскравість пікселів чи інші метадані, які можна отримати з зображення. Інша частина заснована на машинному навчанні.

Задача, що розглядається у даній роботі, полягає в знаходженні за наявними у текстовому документі зображеннями-прикладами схожих на них зображень у базі даних. Пошук відповідностей між зображеннями буде засновано на дескрипторному підході порівняння.

Актуальність задачі полягає у тому, що на даний момент не існує алгоритмів із пошуку плагіату зображень в електронних документах [3].

1.2 Існуючі методи пошуку зображень

Щоб провести пошук якогось об’єкта, необхідно спочатку отримати його характеристику. Це може бути якась з його ознак чи його оточення. При пошуці на основі тексту зазвичай використовують ключові слова вказані у заголовку або в даних сторінки сторінки. Це також можуть бути випадкові фрази, що зустрічаються у документі.

Те ж саме і з зображеннями – є методи пошуку по опису цього зображення, з використанням якихось його метаданих, чи тексту (тегів), які до нього прикріплені [4]. Проте, через більшу складність медіа-даних у порівнянні із текстовими, було створено велику кількість різних алгоритмів пошуку, аби знайти кращий, й ще велика кількість можливих залишається.

1.3 Системи пошуку зображень

Вже існують різні сервіси, які дозволяють виконати пошук зображення за його прикладом чи характеристиками. Серед них: Google Search by Image, TinEye, Yandex Image. Кожен з цих сервісів має окремий підхід до виявлення характеристик зображення й їх подальшого порівняння. Проте, головною метою будь-якого з них є надання користувачу шуканого зображення й (чи) сторінки, де воно з’являється.

1.3.1 Google Search by Image

Google Search by Image – один із сервісів пошуку зображень в інтернеті. Після опрацювання наданих даних, якщо сервіс знайшов зображення, він додасть до результату посилання на ресурс, де воно було знайдене. У якості даних, за якими буде вестися пошук, можна використовувати як теги, так і

контекст, пов’язаний із зображенням. Також можна проводити пошук і по самому зображенню. У цьому разі використовуються його числові характеристики (так званий, «гібридний» підхід).

Треба відмітити таку особливість, як цензурна політику компанії Google. Задля її дотримання, сервіс було налаштовано таким чином, щоб він не додавав до результатів матеріали захищені авторським правом чи інші, які підлягають цензурі. Це налаштування стоїть за замовчуванням, проте його можна відмінити.

1.3.2 TinEye

TinEye – це пошукова система зворотних зображень. Тут йдеться про те, що алгоритм пошуку заснований на числових характеристиках зображення, наданого користувачем й тих, що були вже збереженні у базі даних.

Використовуючи TinEye, можна знайти оригінал зображення, дізнатися чи існують його модифіковані версії, або знайти варіанти з кращою роздільною здатністю. Сервіс дозволяє знайти зображення, навіть якщо воно було геометрично змінене – масштабоване, обрізане. У TinEye не передбачене використання ключових слів, метаданих або водяних знаків.

На відміну від сервісу Google Search by Image, TinEye не гарантує безпеку авторських прав.

1.3.3 Yandex Image

Цей сервіс використовує при пошуку зображення-зразок чи його текстовий опис. При цьому він є стійким до деяких незначних змін цього

зображення. Також, автори додали режим цензури матеріалів. Даний сервіс дуже схожий за функціоналом на сервіс Google Search by Image.

1.3.4 Висновки з огляду систем пошуку зображень

При огляді вищезазначених сервісів, зрозуміло, що вони знаходять плагіат зображення лише за його описом, чи при наданні їм безпосередньо зразку цього зображення. Крім того, це може бути лише один зразок за раз.

Отже, питання обробки кількох зображень та ще й у текстових документах все ще залишається відкритим. Дана робота має на меті об’єднати ці моменти й розробити алгоритм перевірки електронних документів, при якій усі зображення, що вони містять, будуть перевірені на плагіат.

1.4 Огляд методів обробки зображень

1.4.1 Дескриптори ключових точок

У контексті комп’ютерного зору, візуальні дескриптори (або дескриптори зображень) – це описи візуальних особливостей вмісту зображень, чи інших медіа-даних. Під особливостями мають на увазі інформацію про зміст та певні характеристики зображення. Частіше за все, це точка та опис середовища навколо неї: кольорів інших точок, контрастів й переходів. Різні алгоритми дескриптори визначають ці особливості по різному [5].

1.4.1.1 Дескриптори повного циклу

Дескрипторами повного циклу називають алгоритми, що включають у себе алгоритм як із пошуку (детектор) так і з опису (дескриптор) характерних точок зображення.

1.4.1.2 Детектор

Детектор – алгоритм, що знаходить рідкісні характеристики точки у деякій локалізованій зоні зображення. Після опису та порівняння усіх точок на даній ділянці виявляється, чи є точка унікальною. Результат обробки, частіше за все, представляють у вигляді набору координат точок на зображенні, які були виявлені як ключові (характерні).

В детекторів є одна особливість, без якої алгоритм не можна назвати детектором – повторюваність результатів обробки. Тобто, при кожному аналізі одного й того ж зображення (навіть зміненого якимось чином), він буде виділяти на ньому одні й ті ж самі точки кожного разу. Ця особливість є ключовою та дуже корисною при порівнянні зображень.

При визначенні характерних точок зображення, вони можуть бути співвіднесені у різні групи. Деякі з них вже стали еталонними, такі як:

- краї – з’являються у разі зіткнення двох областей на зображенні. Частіше за все, ними є набори точок, у яких виявився великим градієнт, а його структура наближена до одновимірної. Це, зазвичай, зони зіткнення двох поверхонь з різним кольором;
- кути – багато з перших алгоритмів детектування мали на меті саме знаходження кутів предметів на зображенні. Під час пошуку кутів, звертають увагу на рівень градієнту на ділянках зображення з метою знайти такий перепад кольорів, що візуально буде схожий на кут. До цього типу особливостей також відносять «плями». Плями – це невеликі об’єкти, колір

яких різко відмінний від оточуючого (наприклад сонячні зайчики чи плями з під фарби);

− краплі – модернізована версія кутового детектору. На відміну від попередника обробляє не точки, а області. Він так само порівнює характеристики цієї області з оточуючими, як це б робили із точками;

− хребти – до них відносяться витягнуті предмети, такі як дороги. Тобто це не обов'язково прямі. На відміну від пошуку країв, у цьому випадку шукають одновимірні криві й перевіряють їх ступінь ширини.

Виходячи з того, що існують різні способи виділення ознак, а також використовуються різні підходи в обчисленнях характерних точок, існує безліч алгоритмів-детекторів комп'ютерного зору. Нижче наведені деякі з них:

− SUSAN (Smallest Univalue Segment Assimilating Nucles) – відноситься до числа детекторів кутів і країв. Виявляє характерні точки шляхом накладання маски, що перевіряє деяку з характеристик (найчастіше це яскравість) центрального пікселя (ядра) з іншими пікселями під маскою й переносить це значення до окремої матриці. У результаті роботи, коли нова матриця сформована, на основі цих значень вираховується поріг, при задоволенні якого точку вважають характерною;

− Canny edge (кутовий детектор Кєнні) – у наш час є одним із найбільш популярних алгоритмів пошуку характерних точок. Відноситься до групи детекторів, що виявляють краї. При розробці алгоритму, автори приділили особливу увагу врахуванню наступних критеріїв:

1) характерна точка має бути в центрі знайденого краю;

2) в кожного краю має бути лише одна точка, що його характеризує;

3) якщо на зображення накладений шум, він не має впливати на точність;

4) детектор має знайти якомога більше країв, і зробити це правильно.

Для знаходження країв, алгоритм Кєнні використовує градієнти зображення, а після знаходження усіх можливих точок відсіює «слабкі» краї як хибні (до них відносяться точки з малим значеннями градієнту);

− FAST (Features from Accelerated Segment Test) – алгоритм з групи виявлення кутів та крапель. Є одним з найшвидших алгоритмів детектування. В ідеї алгоритму закладено проходження по зображенню колом (плямою) – область, що включає у себе 16 пікселів. Якщо у цій області буде знайдено більше ніж 8 пікселів, що є яскравішими чи темнішими за центральний піксель, ця точка буде вважатися центром кута. Такий підхід до опису точок додає алгоритму стійкості до локальних змін зображення, таких як зміна ракурсу, зміна масштабу, незначні перестановки об’єктів та інше.

1.4.1.3 Дескриптор

Окрім знаходження власне ключової точки, важливо описати цю точку та її оточення, щоб зробити її ще більш унікальною. Як дані для опису, можуть використовуватися колір, текстура, значення оточуючих точок та утворення ними країв, кутів тощо. Звичайно ж, чим більш деталізованим буде опис, тим краще ми зможемо відрізняти одні ключові точки від інших. Проте, це потребуватиме значних ресурсів для обробки й зберігання інформації про ключову точку. З іншого боку, недостатня поглибленість опису даних може негативно позначитися на порівнянні зображень [6]. Алгоритми дескрипторів йдуть на різні компроміси та маніпуляції, задля вирішення цієї проблеми:

− SIFT (Scale-Invariant Feature Transform) – цей алгоритм-дескриптор повного циклу є одним із найбільш популярних у світі. Він використовує локальні характерні точки, що є інваріантними до масштабування й обертання зображення, а також незначних змін ракурсу, шуму та освітлення. І хоча, при такій високій точності роботи, він мав би використовувати великі обсяги ресурсів, насправді цей алгоритм є досить

економним у цьому плані [7]. Простота дескриптору, точність й незначна ресурсоємність зробили цей алгоритм таким популярним;

− SURF (Speeded Up Robust Features) – натхненний послідовний дескриптору SIFT. Також відноситься до локальних дескрипторів повного циклу. Швидший на відміну від свого попередника, проте поступається в точності. Є так само стійким до перетворень й впливів різного роду на зображення. Щоб знайти ключові точки проходить по зображенню маскою (схожий на алгоритм SUSAN, що був описаний вище). При описі характерних точок, використовує оточуючі їх, в деякому радіусі, точки [8];

− ORB (Oriented FAST and Rotated BRIEF) – алгоритм, що відноситься до дескрипторів повного циклу. Його було розроблено як альтернативу дескрипторам SIFT та SURF і, на відміну від них, не було запатентовано, що дозволяє його безкоштовне використання. Дескриптор є вдвічі швидше за свого попередника – SIFT. Й при цьому, здавалося б, він мав бути значно гіршим у точності, проте він виявляє характерні точки майже так само добре як його попередник. Так виходить за рахунок того, що у ньому були об’єднані особливості детектору FAST та дескриптору BRIEF (модифікованої версії – rBRIEF, що є інваріантною до обертань зображення) [9, 10];

− BRIEF (Binary Robust Independent Elementary Feature) – дескриптор повного циклу, особливий тим, що подає опис ключових точок як двійковий вектор з 128-512 бітів. Довжина вектору регулюється у залежності від необхідної точності. На створення біта йде одна ітерація алгоритму, тож, хоча вони й збільшать точність опису, це може зайняти багато часу [11].

1.4.2 Огляд нейронних мереж

Нейронні мережі можна розглядати як сучасні обчислювальні системи, що перетворюють та обробляють інформацію приблизно так само, як

процеси, що відбуваються в мозку людини. Інформація, що буде оброблена, має бути приведена до числового виду, що дозволить користуватися нею нейронній мережі. Так, якщо модель якогось об’єкта має якісні характеристики, вони мають бути переведені у числовий вид, який зрозуміє машина.

Найчастіше нейронні мережі використовують у задачах комп’ютерного зору, таких як розпізнавання, класифікація зображень. Також це може бути класифікація чи аналіз текстової інформації.

Штучні нейронні мережі (ШНМ) є аналогом людського мозку, повторюючи його структуру та спосіб мислення, наскільки це може зробити людина. Головною особливістю людського мозку вважають навчання та самонавчання з використанням досвіду. Це було вдало імплементовано у ШНМ. Системи з використання самоначаємих ШНМ успішно вирішують такі проблеми як: виконання прогнозів, розпізнавання образів, оптимізації.

Алгоритм навчання (який ще називають правилом) – це метод або математична модель, яка є запорукою продуктивності нейронної мережі. При навчанні ШНМ, правило буде багаторазово застосовано до всієї мережі [12]. В результаті його виконання, будуть змінені ваги нейронів мережі. При навчанні мережі, алгоритм використовує поточні ваги мережі, щоб отримати результат і порівняти його з очікуваним (наданим учителем) результатом. Виходячи з результатів цього порівняння та залежно від алгоритму, будуть змінені ваги мережі перед наступною ітерацією навчання.

Під час навчання мереж, використовують різні підходи:

− навчання з учителем (Supervised Learning) – при цьому способі, навчання проводиться на двох векторах – приклади значень, які має бути навчена обробляти система, та вихідні дані (відповіді) до кожного з них. Ці вектори прийнято називати навчальною парою, а множину навчальних пар – навчальною вибіркою. Навчання складається із почергового подання пар до системи ваг і отримання результату, порівняння його із фактичною

відповіддю із вектору вихідних даних та зміні вагів мережі, аби виправити похибки;

− навчання без учителя (Unsupervised Learning) – інший спосіб навчання ШНМ. Під час навчання цим способом, мережа хаотично навчається виконувати поставлене завдання. Будь-яке втручання до результатів роботи системи заборонене. Цей підхід працює, коли є навчальна вибірка, у якій необхідно виявити внутрішні взаємозв'язки, певні залежності чи закономірності;

− навчання з підкріпленням (Reinforcement Learning) – поєднання двох попередніх варіантів. Роль «учителя» заміняє «критик». Він слідкує за роботою системи, перевіряє реакції середовища на вхідні сигнали та визначає похибки, після ітерацій навчання [13].

1.4.3 Кластеризація

Кластеризація (кластерний аналіз) – процес розбиття деякої кількості об'єктів на групи (кластери). Кожен кластер повинен містити схожі, за деякими встановленими, або вирахуваними, ознаками, об'єкти. В той самий час, різні кластери мають містити максимально різні об'єкти. Від класифікації, кластеризація відрізняється тим, що перелік груп чітко не заданий й у процесі роботи алгоритму.

1.4.3.1 Методи кластеризації

До типових методів кластеризації відносять:

− ієрархічні – мають на меті групування набору даних за допомогою використання ієрархічного дерева кластерів;

− центроїдні – в них кластер представляють деяким одним значенням, яке є його центром;

− статистичні – в них кластери будуються за допомогою статистичних розподілів;

− засновані на щільності – відносить до одного кластеру точки, які розташовані найбільш щільно, точки що лежать поодиноко можуть бути помічені як викиди;

− на основі сітки – цей тип підходу до кластеризації стосується не самих точок даних, а простору значень, який їх оточує. Алгоритм зазвичай має наступні кроки: створити структуру сітки, обчислити щільність комірки для кожної з клітинок сітки, класифікувати дані відповідно до щільності клітинок, визначаючи, центри кластерів і перевіряючи поперечний переріз сусідніх клітин [14].

Крім того, розрахунок відстаней між елементами та центрами кластерів теж виконується багатьма способами.

1.4.3.2 Міри відстаней

Найчастіше будуються вектори на основі числових характеристик, проте існують навіть алгоритми, що працюють з текстовими значеннями [15].

Основні метрики:

− Евклідова відстань – найбільш поширена метрика, являє собою корінь суми квадратів різниць значень елементів:

$$\sqrt{\sum_{i}^{n}(x_{1i} - x_{2i})^2};$$

– квадрат Евклідової відстані – власне квадрат попередньої метрики. За рахунок такої зміни, значення більш віддалених точок стає вагомішим:

$$\sum_{i}^{n}(x_{1i} - x_{2i})^2;$$

– Мангеттенська відстань – результати є схожими до метрики евкліда, проте значення у разі великих різниць характеристик не таким великим:

$$\sum_{i}^{n}\left|x_{1i} - x_{2i}\right|;$$

– відстань Чебишева – максимізує різницю між значеннями елементів:

$$max_{i=1\ldots n}(\left|x_{1i} - x_{2i}\right|;$$

– ступінна відстань – метрика евкліда, де користувач сам може визначити ступені, аби змінити вплив різниці значень:

$$\sqrt{\sum_{i}^{n}(x_{1i} - x_{2i})^p}.$$

При кластеризації, залежно від потреби, можна виділяти не лише задачу віднести дані до якогось з кластерів, а й відношення даних з різних кластерів.

1.4.3.3 Міри відношень кластерів

Часто, трапляються дані, що не можна віднести до кластера взагалі. Чи такі, що підходять до кількох кластерів за різними критеріями, або ж різниця значень критеріїв незначна.

У таких випадках розширюють поведінку алгоритму й кластеризації можна розділити наступним чином:

- жорстка кластеризація – кожен об’єкт має належати кластеру не належати до нього;
- м’яка кластеризація – кожен об’єкт належить кожному кластеру з певною вірогідністю.

Серед них виділяють ще декілька різновидів:

- жорстке розбиття на кластери – кожен об’єкт має належати тільки одному кластеру й ніяк інакше;
- жорстке розбиття на кластери з викидами – об’єкт може не належати жодному кластеру й, у такому разі, розглядається як викид;
- кластери з перетином – об’єкт може належати більш ніж одному кластеру в повній мірі;
- ієрархічна кластеризація – існують кластери предки та нащадки. Усі об’єкти нащадку належать до кластеру предка.

У випадках, коли використовуються ієрархічні кластери, постає ще одне питання – як вирахувати відстань між ними [16].

1.4.3.4 Відстань між кластерами

Існує декілька метрик розрахунку відстаней між кластерами:

- одиничний зв’язок – береться відстань між двома найближчими елементами цих кластерів;

− повний зв’язок – береться два найвіддаленіші елементи кластерів. Проте, ця відстань має сенс лише у випадку, коли форма кластерів не є розтягнутою;

− не зважене попарне середнє – розраховується середня відстань від усіх відстаней пар об’єктів цих кластерів;

− зважене попарне середнє – відмінний від попереднього тим, що враховує можливість різної кількості елементів у кластерах;

− незважений центроїдний метод – відстань розраховується безпосередньо між центрами кластерів;

− зважений центроїдний метод – відмінний від попереднього тим, що враховує можливість різниці у розмірах кластерів.

1.4.4 Аналіз спотворень зображень під час збереження в текстових файлах

Якість зображень дуже важлива під час роботи з дескрипторами і, особливо, при їх порівнянні. Під час роботи з оригіналами зображень це, скоріше за все, не викличе проблем (найгірше, що може трапитися – буде змінено формат зображення). Проте, під час аналізу текстових документів, було зроблено висновки, що існує ряд змін зображень, що зустрічаються найчастіше й можуть вплинути на результати їх обробки [17].

1.4.4.1 Геометричні спотворення зображень

Доволі часто, зображення, що використовуються у документах, текстових редакторах тощо, втрачають свій оригінальний вигляд через зміни, необхідні автору. До цих змін можна віднести такі:

− відсічення частин зображення;

- повороти;
- масштабування;
- накладання на зображення тексту чи інших елементів.

У наукових роботах найчастіше зустрічається масштабування оригінального зображення. Рідше можна спостерігати відсічення чи локальні зміни, такі як зміна мови чи тексту, видалення певних частин зображення (не відсічення сторін). Попри те, що ці зміни можуть здаватися незначними, вони можуть вплинути на результати порівняння. Тож, для коректного їх оброблення необхідно використовувати спеціальні алгоритми [18].

1.4.4.2 Збереження зображень у Word документах

Під час зберігання зображень у текстових документах часто виконується автоматичне стиснення. Так, наприклад, текстовий редактор Word Office за замовчуванням має встановлений параметр стиснення зображень до формату 220ppi (пікселів на дюйм). Для користувача це не є дуже помітним, проте для програми може створити ускладнення. Інша кількість пікселів, їх значення, розташування при числовому описі характеристик зображення, зокрема виявлення дескрипторів, можуть призвести до неправильного їх опису. Ці зміни не відносяться до змін формату файлу зображення, тож найвірогідніше можуть бути віднесені до масштабування.

Такі зміни можна спостерігати не тільки у текстовому редакторі Word, але й у багатьох інших, що не представлені у даній роботі. Це зумовлено тим, що кожен з редактор намагається стиснути вкладені зображення, щоб уникнути обтяження файлу.

Хоча документації редакторів кажуть лише про зміни роздільної здатності зображень, слід все ж таки відмітити ймовірність конвертації формату зображення. Бо в цьому разі є ризик втратити пікселі чи отримати

шум на зображенні, який виникає під час зміни формату з більш точного у менш точний.

1.5 Огляд додатків для роботи із зображеннями

1.5.1 Бібліотеки

1.5.1.1 OpenCV

OpenCV – відкрита бібліотека комп’ютерного зору. У ній міститься набір функцій для роботи з медіа-даними та необхідних, для цих функцій, типів даних. Зокрема, у ній зібрано наступні алгоритми для обробки зображень:

- розпізнавання об’єктів;
- визначення характеристик об’єкту (наприклад форми);
- розпізнавання жестів;
- відстеження руху певних об’єктів;
- розпізнавання облич;
- усунення оптичних недоліків;
- співставлення зображень для виявлення спільних об’єктів;
- перетворення зображень;
- обробка та аналіз зображень;
- реконструкція об’єктів та інші.

1.5.1.2 TensorFlow

TensorFlow – бібліотека для машинного навчання з відкритим кодом, створена для автоматичного аналізу та класифікації зображень. Вона була розроблена компанією Google як система машинного навчання. Після надання її у вільний доступ стала часто використовуватися для розрізнання об’єктів на

зображеннях з подальшою їх анотацією. Заснована на архітектурі паралельних обчислень CUDA (архітектура паралельних розрахунків розроблена NVIDIA, у якій використовується графічний процесор), має дуже високу ефективність у плані швидкості обробки.

1.5.1.3 Caffe

Caffe – відкрите середовище глибинного навчання. Caffe найчастіше використовується для сегментації зображень, з подальшою їх класифікацією. Середовище використовує готові та вже перевірені алгоритми нейронних мереж, серед них такі відомі як GoogleNet, AlexNet.

1.5.1.4 MobilNetV2

Keras (відкрита нейромережна бібліотека, написана мовою Python) містить ряд попередньо навчених мереж («додатків»), які можна завантажити та використовувати одразу. Одним із них є MobileNetV2, який навчено класифікувати зображення.

MobileNetV2 можно просто імпортувати із keras.applications і створити його екземпляр. Мережа буде завантажена й готова до роботи. Її можна перенавчити, залежно від потреб користувача.

Якщо не передати жодних ключових слів у MobileNetV2, тоді мережа матиме випадкові ваги; тобто буде створено архітектуру мережі, але не ваги, тому доведеться повністю навчати її самостійно. Параметр weights=’imagenet’, вказує, що необхідно попередньо навчити мережу. В такому разі, достатньо буде змінити вихідний шар – виведення результатів класифікації.

Поява MobileNet вже сама по собі зробила революцію в комп’ютерному зорі на мобільних платформах. MobileNetV2 стала наступним поколінням нейромереж цього сімейства. Вона дозволяє досягати приблизно такої ж точності розпізнавання, як її попередня версія, та ще більшої швидкості роботи.

Основний будівельний блок цієї мережі загалом схожий на попереднє покоління, але має низку ключових особливостей.

Як і в MobileNetV1, тут є блоки згортки з кроком 1 і з кроком 2. Блоки з кроком 2 призначені для зниження просторової розмірності тензора і, на відміну блоку з кроком 1, не мають залишкових зв’язків.

Блок MobileNet, званий авторами розширюючим блоком згортки (в оригіналі expansion convolution block або bottleneck convolution block with expansion layer), складається з трьох шарів:

- спочатку йде pointwise convolution з великою кількістю каналів, званий expansion layer. Цей шар створює відображення вхідного тензора у просторі великої розмірності. Автори називають таке відображення «цільовим різноманіттям» (в оригіналі «manifold of interest»);
- потім йде depthwise convolution з ReLU6-активацією;
- наприкінці цього шару йде 1×1-згортка з лінійною функцією активації, що знижує кількість каналів. Автори висувають гіпотезу, що «цільове різноманіття» високої розмірності, отримане після попередніх кроків, можна «вкласти» в підпростір меншої розмірності без втрати корисної інформації, що, власне, і робиться на цьому кроці (як можна побачити за експериментальними результатами, ця гіпотеза повністю виправдовується).

1.5.2 Мови програмування

1.5.2.1 Python

Python – високорівнева мова програмування загального призначення. Її головна мета – збільшення продуктивності роботи розробника, чистота та читабельність коду. Python часто використовують при написанні програм з машинного зору чи нейронних мереж. Так сталося через велику кількість бібліотек та фреймворків написаних для цієї мови. До них відносяться й велика кількість рішень з питань комп'ютерного зору. Окрім того, у мережі Інтернет можна знайти велику кількість різних статей та блогів з Python на різні теми. Велика кількість статей з написання програм з комп'ютерного зору полегшує вивчення питання та пришвидшує написання коду, що є значною перевагою.

1.5.2.2 Java

Java – строго типізована об'єктно-орієнтована мова програмування. Її головна особливість – інтерпретація коду у так званий байт код. Ця особливість дозволяє робити програми незалежними від операційної системи. В Java є реалізація бібліотеки OpenCV для роботи з зображеннями. На жаль, вона не є такою розвиненою та не має настільки великого числа додатків до неї як Python. Але незважаючи на це, вона є більш продуктивною та швидкою, якраз через відсутність цієї великою кількості налаштувань, що в результаті уповільнює її. Реалізація OpenCV на Java містить усі базові й необхідні алгоритми комп'ютерного зору, що робить її гарним претендентом для написання програм з обробки зображень.

1.5.3 База даних

Для збереження даних у ході даної роботи, було обрано базу даних – PostgreSQL. PostgreSQL – це вільна об’єктно-реляційна система управління базами даних (СУБД).

Сильними сторонами PostgreSQL вважаються:

- високопродуктивні та надійні механізми транзакцій та реплікації;
- успадкування;
- можливість індексування геометричних (зокрема, географічних) об’єктів і наявність розширення PostGIS, що базується на ній;
- вбудована підтримка слабоструктурованих даних у форматі JSON з можливістю їхньої індексації;
- система вбудованих мов програмування, що розширюється: у стандартній поставці підтримуються PL/pgSQL, PL/Perl, PL/Python і PL/Tcl; додатково можна використовувати PL/Java, PL/PHP, PL/Py, PL/R, PL/Ruby, PL/Scheme, PL/sh та PL/V8, а також є підтримка завантаження модулів розширення мовою C;
- розширюваність (можливість створювати нові типи даних, типи індексів, мови програмування, модулі розширення, підключати будь-які зовнішні джерела даних).

1.6 Постановка задачі дослідження

Таким чином, пошук плагіату зображень в електронних документах є актуальним завданням обробки та розпізнавання зображень. Тому завдання це розробка алгоритму знаходження підозрілих на плагіат зображень у текстових документах з кластеризацією їх у базі даних з метою полегшення пошуку.

Об’єктом дослідження є питання кластеризації зображень, для найбільш точного їх віднесення до конкретної групи.

Метою дослідження є розробка та дослідження методу виявлення підозрілих на плагіат зображень в електронних документах. Особливу увагу необхідно приділити швидкості пошуку зображень за рахунок попередньої кластеризації зображень в електронній колекції [19].

Задля досягнення цієї мети необхідно:

- вивчити питання кластеризації зображень;
- дослідити питання поділу зображень на класи на основі нейромережевого підходу;
- ознайомитися із використанням навченних заздалегідь нейронних мереж;
- дослідити використання гістограм у якості ознак зображень для подальшого етапу кластеризації;
- розробити поетапний класифікатор для аналізу вхідних зображень;
- сформувати датасет для перенавчання моделі нейронної мережі;
- спроєктувати та розробити застосунок для пошуку плагіату зображень в електронних документах;
- сформувати базу даних зображень для дослідження роботи застосунку;
- провести дослідження швидкості та якості роботи застосунку.

2 РОЗРОБКА ТА ДОСЛІДЖЕННЯ МЕТОДУ ДЛЯ ПОШУКУ ЗОБРАЖЕНЬ, ПІДОЗРІЛИХ НА ПЛАГІАТ

2.1 Опис класифікації на основі нейронної мережі

2.1.1 Згорткові нейронні мережі

Штучна нейронна мережа (ШНМ) – математична модель та її апаратне чи програмне втілення, побудоване за принципом організації та функціонування біологічних нейронних мереж. ШНМ виглядає як система з’єднаних і взаємодіючих між собою простих процесорів (штучних нейронів).

Згорткова нейронна мережа (ЗНМ) – спеціальна архітектура штучних нейронних мереж, націлена на ефективне розпізнавання образів, входить до складу технологій глибокого навчання. ЗНМ будуються на трьох базових ідеях: локальне рецептивне сприйняття, колективні ваги (ядро згортки) (рис. 2.1), субдискретизація. ЗНМ використовують різновид багатошарових перцептронів, розроблений так, щоб вимагати використання мінімального обробки. Це досягається завдяки їхній архітектурі спільних ваг та характеристик інваріантності відносно паралельного перенесення [20].

ЗНМ використовують порівняно мало попередньої обробки, в порівнянні з іншими алгоритмами класифікування зображень. Це означає, що мережа навчається за допомогою фільтрів, що в традиційних алгоритмах приходиться розробляти вручну. Ця незалежність у конструюванні ознак від апріорних знань та людських зусиль є великою перевагою.

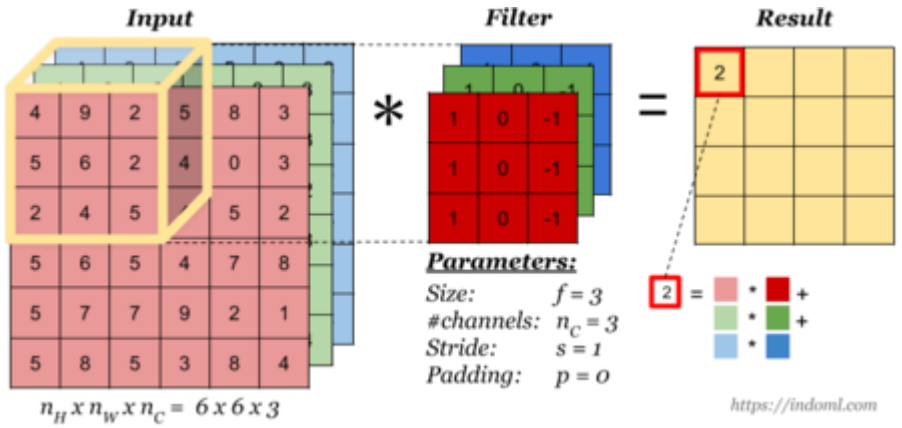


Рисунок 2.1 – Приклад згортки с трьохвимірним ядром

2.1.2 Використання навченних нейронних мереж

Найбільшою проблемою, при навчанні нейронної мережі, є необхідність у великих обчислювальних потужностях обладнання й великих обсягах даних, на яких цю мережу будуть навчати.

Щоб уникнути цих проблем, стали розповсюджувати навчені нейронні мережі. Вони вже є навченними на величезних обсягах різних видів даних (залежно від обраної мережі). Користувачу необхідно лише змінити останній шар моделі – вихідний, щоб результати класифікації за допомогою мережі вкладалися у обрану користувачем кількість класів [21]. На рисунку 2.2 цей шар представленний як «Output classes».

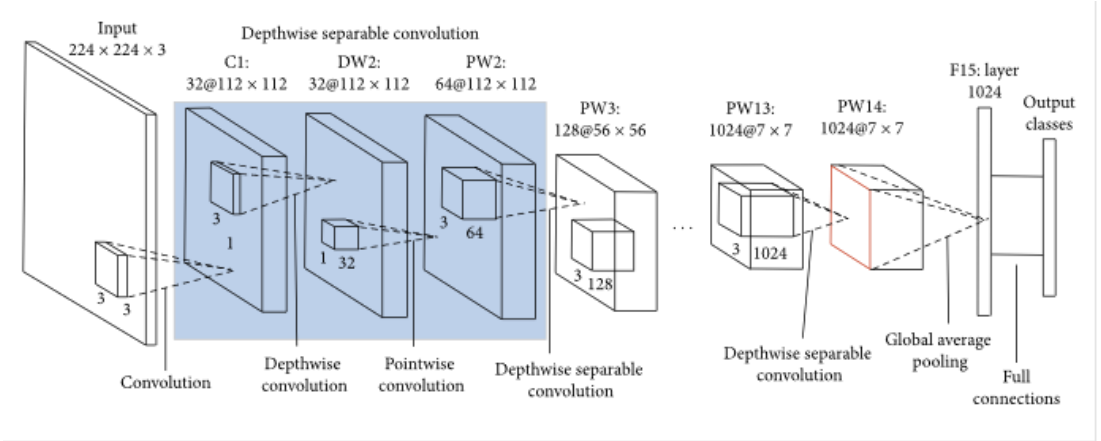


Рисунок 2.2 – Представлення навченної нейронної мережі Mobilenet v2

Також, ці моделі можна донавчити. Якщо не відключати перерозподіл ваг моделі, вона буде формувати їх на новому наборі даних, які користувач передасть їй. Якщо ж «заморозити» модель, то навчатися буде лише останній шар, який при першому використанні буде заповнений випадковими значеннями. Звісно, краще навчати лише останній шар моделі, оскільки при значення останнього шару не будуть одразу давати точний результат і це може сильно змінити усі ваги не «замороженої» моделі.

2.1.3 Формування датасету

Хоча модель є навченною на великому обсязі даних, все одно треба ще навчити останній шар, бо при його зміні, задля відповідності кількості класів, що будуть використані у роботі, його значення ще не зважені.

Для цього необхідно створити, або знайти, інший датасет зображень, які підходять обраній предметній області, й провести навчання на них. Чим більшим є навчальна вибірка, тим більш точною буде навчена модель.

В даному випадку, предметна область – наукові роботи й відповідно очікується два класи зображень:

- схеми, діаграми, графіки та інше;
- будь-які фотозображення, наприклад природи, міст, людей, тварин та інше.

2.2 Розробка критерію для віднесення зображення до підозрілих щодо плагіату

Задача пошуку зображень, що схожі на плагіат, може мати таку постановку. Нехай існує множина M зображень B_i, i=1,…,$|M|$, що знаходиться

у БД, та зображення-зразок пошуку B_0. Необхідно сформувати підмножину зображень $M''' \subset M$, для якої виконується визначений критерій [22].

В роботі пропонується використовувати багатоетапний критерій, який буде складатися із трьох перевірок зростаючої складності. Оскільки більш складні перевірки потребують більше часу, необхідно простими перевірками відкинути зображення, які зовсім не схожі на плагіат, що дозволить скоротити загальний час пошуку.

Було прийнято рішення, проводити таке фільтрування, базуючись на гістограмі кольорів зображення, після переведення його у формат відтінків сірого [23, 24]. Усі можливі варіанти яскравості (0-255) пікселів буде поділено на 10 груп, так щоб охопити весь спектр кольорів. При збережені зображення у базу, буде також збережено ці дані, задля пришвидшення подальших аналізів.

Критерій 1. Зображення з множини *M* буде обраним для подальшого порівняння, якщо міра схожості гістограм задовольняє умові

$$\rho(G_0,\ G_i) \leq t', \tag{2.1}$$

де $G_0 = (g_{0_1}, \dots, g_{0_N})$, $G_i = (g_{i_1}, \dots, g_{i_N})$ – гістограми яскравості зображення-зразка пошуку B_0 та зображення із БД B_i відповідно;

N – кількість відліків гістограми, в роботі *N*=10;

ρ() – міра схожості гістограм, яка обчислюється як нормована Манхетенська відстань

$$\frac{\sum_{j=1}^{N} |g_{o_j} - g_{i_j}|}{N*S},$$

де *S* – розмір зображення (width*height);

t’ – поріг для порівняння гістограм, що обраний експериментально, в роботі *t*’=0,1.

За таким значенням t’, скоріше за все зображення є схожим, принаймні за кольоровим спектром, а якщо в оригіналу зображення була змінена яскравість, воно не буде відсіяним.

Із зображень, що відповідають критерію 1 (2.1) буде сформована підмножина $M' \subset M$, яка буде далі перевірена на відповідність критеріям 2, 3.

У разі, якщо підмножина M' є порожньою (жодне з зображень бази не відповідає критерію 1), надане на перевірку зображення B_0 вважається не плагіатним.

Наступним кроком, для зображень із M' буде перевірено кількість знайдених відповідностей [25-27], оскільки у разі порівняння дуже різних зображень їх кількість може бути занадто малою задля подальших перевірок й, через специфіку наступного критерію, постійно істинними. При цьому мати на увазі можливість зменшення числа цих пар через геометричні зміни, чи малий розмір зображень.

Експериментальним шляхом, було встановлено, що необхідна мінімальна кількість пар точок, що має бути знайдена при аналізі зображень – 40. У такому разі відсіюються завше не відповідні зображення й ще не відкидаються ті, які є схожими, але не мають великої кількості характерних точок.

Критерій 2. Зображення з множини $M' \subset M$ буде обраним для подальшого порівняння, якщо кількість знайдених відповідностей методом NNDR задовольняє умові

$$NM > t'', \tag{2.2}$$

де NM (number of matches) – кількість усіх відповідностей, виявлених методом NNDR;

t'' – поріг для перевірки кількості NM, у роботі *t''*=40.

В результаті перевірки умови критерію 2 (2.2) буде сформована підмножина зображень $M'' \subset M'$.

Останнім критерієм перевірки зображення на плагіат було обрано порівняння кількості пар ознак, виявлених як відповідні, при обробці методами NNDR і RANSAC [28, 29]. У тих зображень, які не є відповідними, після обробки множин пар точок буде відсіяно велику кількість тих, що не є істинними. У такому разі, можна сказати, що коли після відсіювання хибних точок, у нас залишаться досить велика їх кількість, у порівнянні з початкової, то зображення скоріше за все є істинним.

Експериментально, було встановлено, що оптимальна кількість таких точок становить не менше 75% від початкової кількості точок з відповідностями.

Критерій 3. Зображення з множини $M'' \subset M'$ буде обраним як підозріле на плагіат, якщо співвідношення кількість відповідностей, отриманих методом RANSAC, до кількості відповідностей, що знайдено методом NNDR, задовольняє умові

$$\frac{NI}{NM} > t''', \tag{2.3}$$

де NI (number of inliers) – кількість відповідностей, що були визнані вірними методом RANSAC;

NM (number of matches) – кількість усіх відповідностей, виявлених методом NNDR;

t''' – поріг для перевірки частки відповідностей, отриманних методом RANSAC, у роботі *t'''*=0,75.

В результаті перевірки умови критерію 3 (2.3) буде сформована підмножина зображень $M''' \subset M''$, пошук якої є кінцевою метою пошуку підозрілих на плагіат зображень.

2.3 Формування ознак зображень на основі гістограм з урахуванням положення пікселів

Гістограма – це графік статистичного розподілу елементів цифрового зображення з різною яскравістю, в якому по горизонтальній осі представлена яскравість, а по вертикалі – відносна кількість пікселів з конкретним значенням яскравості.

Гістограму можна побудувати як для монохромного, так і для кольорового зображення [30, 31]. У випадку із кольоровими зображеннями можна будувати окрему гістограму для кожного окремого каналу, або усереднити їх значення.

Оскільки, у даній роботі було вирішено використовувати гістограму для попередньої обробки зображень, постає наступна проблема: для фотозображень з великою кількістю різноманітних пікселів такий підхід працюватиме добре, але для схем, графіків, тощо, де багато вільного простору, може виникнути похибка. Адже при опрацюванні гістограми, не враховується положення пікселів, лише їх кількість і це стане проблемою, у разі якщо два графіки відрізняються лише чвертю на осях координат.

Для уникнення цієї проблеми, було вирішено розбивати кожне зображення на 4 сектори і рахувати гістограму окремо для кожного сектору. Потім, результати буде об’єднано в один вектор, який буде визначним для зображення. За допомогою нього й проходитиме подальше порівняння.

2.4 Порівняння на основі дескрипторів

Після знаходження дескрипторів зображень, необхідно їх порівняти, аби визначити, чи є зображення відповідними одне одному. Проте, точність знайдених відповідностей, особливо при порівнянні різних зображень, може

бути достатньо низкою [32]. Для збільшення точності знайдених відповідностей потрібно відсіяти ті з них, які є хибними.

2.4.1 Метод *k* найближчих сусідів та його модифікація NNDR

KNN (*k* найближчих сусідів) – метод машинного навчання, метою якого є на великому наборі навчальних даних знайти *k* найближчих (найбільш відповідних) новим даним, що надійшли до системи. Число *k* вираховується як квадратний корінь від загального числа даних у базі [33, 34].

Проте, у нашому випадку буде виконуватися знаходження лише двох найближчих дескрипторів, відносно шуканого. Це необхідно задля подальшої обробки хибних співвідношень методом NNDR. Як видно з рисунку 2.3, метод KNN знаходить дуже багато віповідностей, частина з яких є хибними.

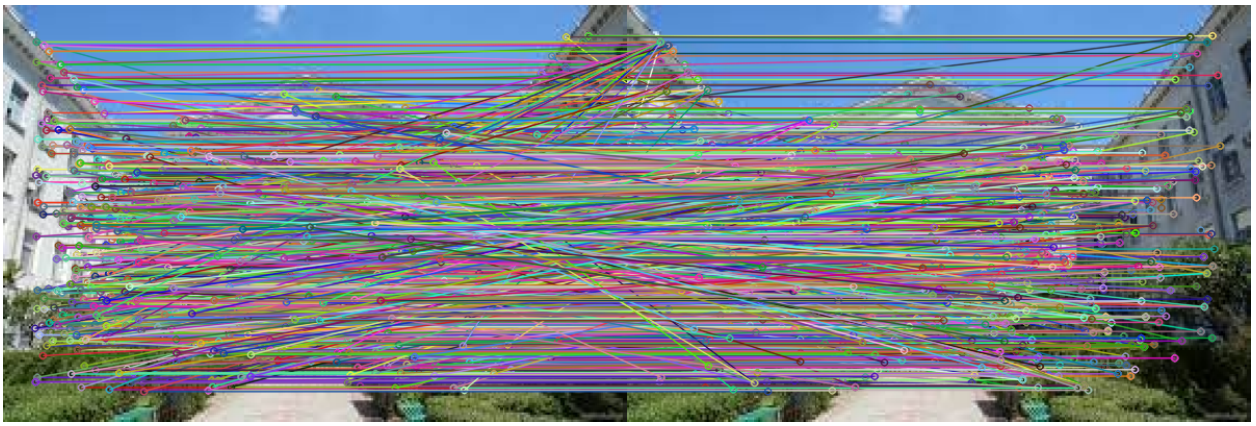


Рисунок 2.3 – Результат знаходження пар характерних точок методом KNN (2231 відповідностей, дескриптор SIFT)

NNDR (співвідношення відстані найближчих сусідів) – є модифікацією від KNN, головною ідеєю якого є пошук одного найближчого сусіда (найвідповідніших даних з набору) до даних, що надійшли на перевірку. При порівнянні дескрипторів двох зображень, він використовується аби попередньо відсіяти хибні відповідності.

Суть алгоритму полягає у тому, що для характерної точки еталонного зображення B_0 обирають дві точки на зображенні B_i, що порівнюється з еталонним, які мають найбільш схожі дескриптори [35]. Якщо відношення евклідових відстаней дескрипторів більше за деякий поріг (2.4), то пара точок (точка з еталонного зображення та точка з найбільш схожим дескриптором порівнювального зображення) є дійсною. В іншому разі, вважається, що дана характерна точка з еталонного зображення не має відповідної точки у другому зображені. Тобто повинна виконуватися умова

$$\frac{\rho(D_{0_l}, D_{i_{nearest\_1}})}{\rho(D_{0_l}, D_{i_{nearest\_2}})} \leq \sigma, \qquad (2.4)$$

де ρ( ) – міра схожості дескрипторів з зображень B_0 та B_i, яка обчислювалась як евклідова відстань, причому $\rho\left(D_{0_l}, D_{i_{nearest\_1}}\right) < \rho\left(D_{0_l}, D_{i_{nearest\_2}}\right)$;

D_{0_l} – дескриптор характерної точки еталонного зображення B_0;

$D_{i_{nearest\_1}}$, $D_{i_{nearest\_2}}$ – дескриптори характерних точок зображення B_i, які мають дві найменші значення;

σ – задана величина точності, яка в роботі мала значення σ=0,75).

Як видно із рисунку 2.4, навіть після обробки набору дескрипторів методом NNDR залишаються хибні пари ознак. Задля їх усунення, зображення буде оброблено додатково методом RANSAC.

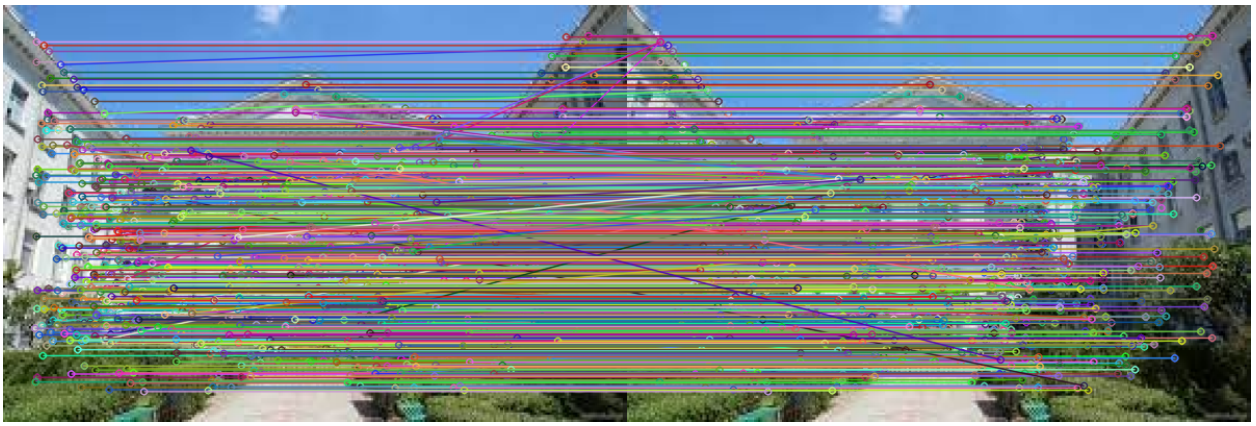


Рисунок 2.4 – Результат відбору вірних пар дескрипторів методом NNDR (1889 відповідностей, дескриптор SIFT)

2.4.2 Видалення хибних відповідностей та пошук параметрів геометричного перетворення між зображеннями методом RANSAC

Метод RANSAC є більш точним, проте набагато більш трудомістким, у порівнянні з NNDR. Ідеєю алгоритму є знаходження функції перетворення зображення, за допомогою якої з одного зображення (того, що перевіряють) можна отримати інше (те, з яким порівнюють) на основі їх дескрипторів [36].

Оскільки мова йде про застосування до усіх характерних точок із відповідностями деяких змін, то чим менше точок надійде до цього алгоритму, тим швидшим буде опрацювання [37-39]. Для цього й необхідно попередньо відсіяти завчасно хибні точки методом NNDR.

RANSAC є ітераційним методом. Під ітерацією мається на увазі побудова функції та аналіз кількості точок, які дійсно будуть переведені з одного зображення у інше, використовуючи цю функцію. Сама функція будується на основі кількох, випадково обраних точок, таким чином, аби вони точно були правильно переведені.

З усіх ітерацій, обирається функція, за якої найбільша кількість відповідностей характерних точок є вірною. Усі точки, що не були переведені у відповідні їм за цією функцією, відсіюються як хибні. Оскільки дані для

побудови функції на кожній ітерації випадкові, не можна точно визначити необхідну їх кількість задля отримання найбільш точних результатів [40]. Таким чином, точність роботи алгоритму зростає разом із кількістю проведених ним ітерацій. Результати роботи методу RANSAC представлено на рисунку 2.5.

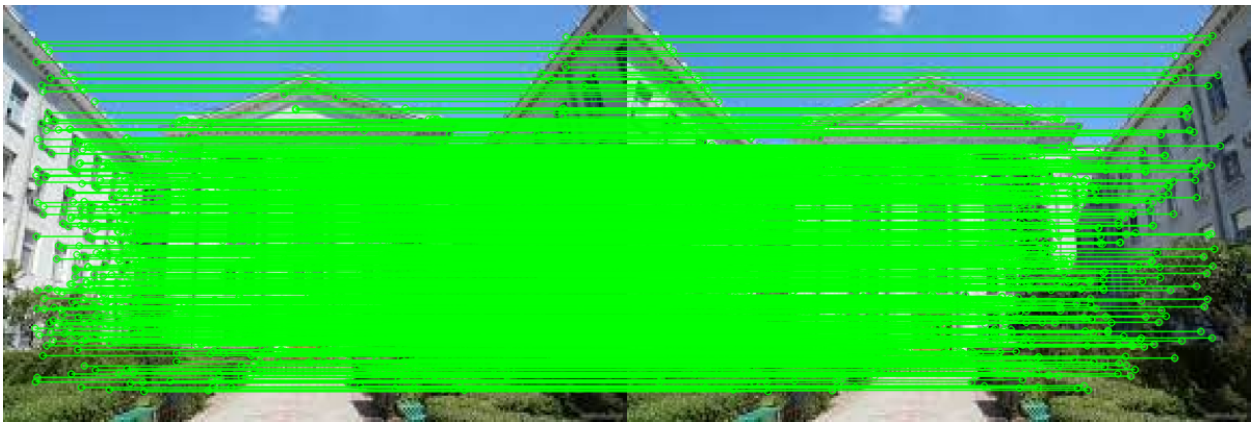


Рисунок 2.5 – Результат обробки пар дескрипторів методом RANSAC (1607 відповідностей, дескриптор SIFT)

2.5 Розробка алгоритму пошуку зображень, що є підозрілими на плагіат

На основі результатів, отриманих у результаті попередньої (бакалаврської) роботи, розроблених тоді критеріїв та нових, внесених змін, які мають на меті покращити швидкість і точність роботи застосунку, маємо такий алгоритм роботи програми:

Крок 1. До програми відправляється запит, який містить у собі текстовий документ із зображеннями.

Крок 2. Програма аналізує текстовий документ й витягує з нього файли зображень й починає аналіз першого зображення.

Крок 3. Зображення аналізується навченною нейронною мережею, щоб дізнатися основний його клас: фотозображення чи схема.

Крок 4. У зображення виявляються дескриптори характерних точок, будується його гістограма.

Крок 5. На основі гістограми зображення, по базі даних виконується пошук схожих зображень на основі критерію (2.1). У разі коли таких зображень немає, програма переходить до Кроку 9. Інакше – до Кроку 6.

Крок 6. Проводиться пошук пар відповідних дескрипторів за допомогою методів KNN і NNDR. У разі, якщо не виконується критерій 2 (критерій мінімальної кількості пар дескрипторів) (2.4), знайдених NNDR, програма переходить до Кроку 9. Інакше – до Кроку 7.

Крок 7. Набір пар відповідних ознак просіюється методом RANSAC. За критерієм 3 (2.3) оцінюється виконання критерію кількості хибних пар. Якщо критерій не виконується, програма переходить до Кроку 9. Інакше – до Кроку 8.

Крок 8. Зображення пройшло перевірку усіх критеріїв та є підозрілим на плагіат. Його додано до списку таких зображень, що буде подано на вихід програми. У разі, якщо є не перевірені зображення з файлу, програма переходить до наступного й починає його перевірку з Кроку 3.

Крок 9. Якщо зображення не пройшло одну з перевірок критеріїв, воно не є підозрілим на плагіат. Його буде додано до бази даних, як еталон для подальших перевірок. У разі, якщо є не перевірені зображення з файлу, програма переходить до наступного й починає його перевірку з Кроку 3.

Крок 10. Якщо зображень, які необхідно перевірити, не залишилось, програма подає на вихід список зображень, які були виявлені підозрілими на плагіат та завершує свою роботу.

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ МЕТОДУ ПОШУКУ ЗОБРАЖЕНЬ, ПІДОЗРІЛИХ НА ПЛАГІАТ

3.1 Опис датасету

Задля навчання нейронної мережі та тестування роботи програми було створено два датасети зображень.

Перший датасет (Coco) містить 600 фотозображень, які включають у себе фото природи, будинків, арт об’єктів, різних за розміром предметів тощо, узятих із відкритого репозиторію фотозображень COCO (рис. 3.1).

Другий датасет (Schemas) містить 600 зображень схем та різних діаграм, які часто використовуються у наукових роботах (рис. 3.2).

Усі зображення було приведено до розміру 224×224 пікселі для навчання на них нейронної мережі. До того ж, перед навчанням кожне з них було піддане нормуванню яскравості, для більш точного навчання мережі. Як навчальні вибірки, було використано по 550 зображень з кожного датасету, інші ж 50 зображень з кожного датасету використовувалися у тестуванні застосунку.

Для тестування програми, випадкові зображення з обох датасетів зберігалися у текстовому файлі. До деяких із них було застосовано геометричні зміни (здебільшого однорідне й неоднорідне масштабування), деякі залишились без змін.

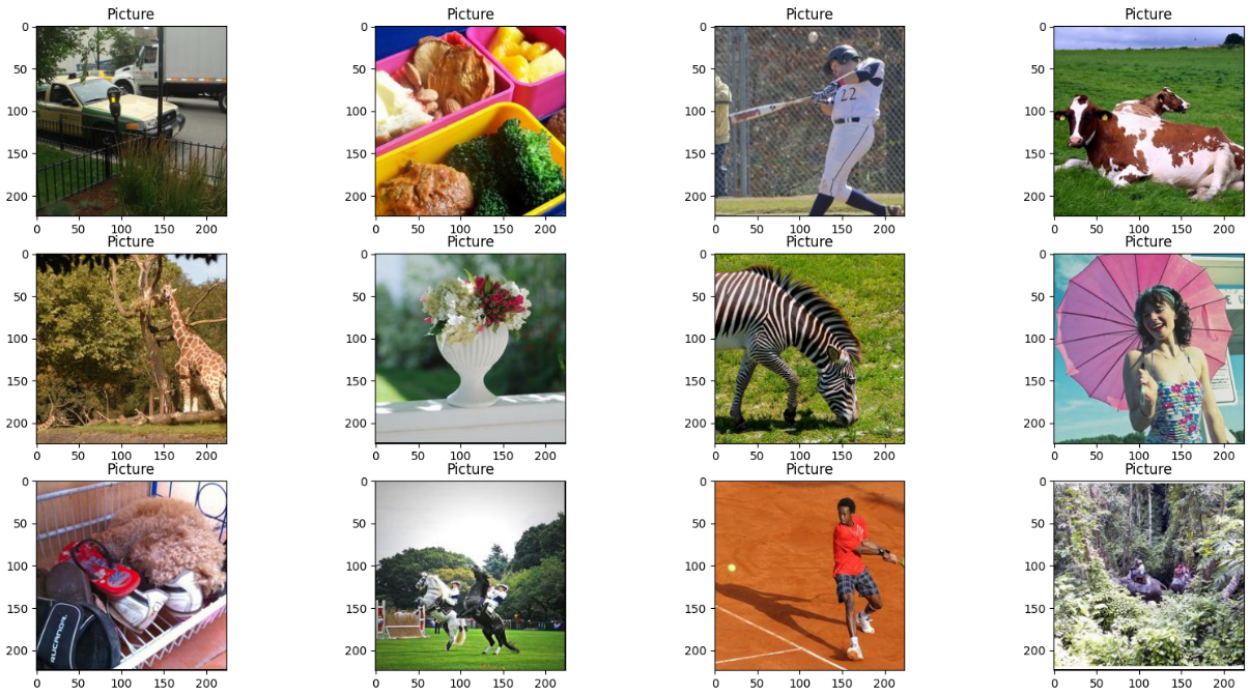


Рисунок 3.1 – Приклади зображень із датасету Coco

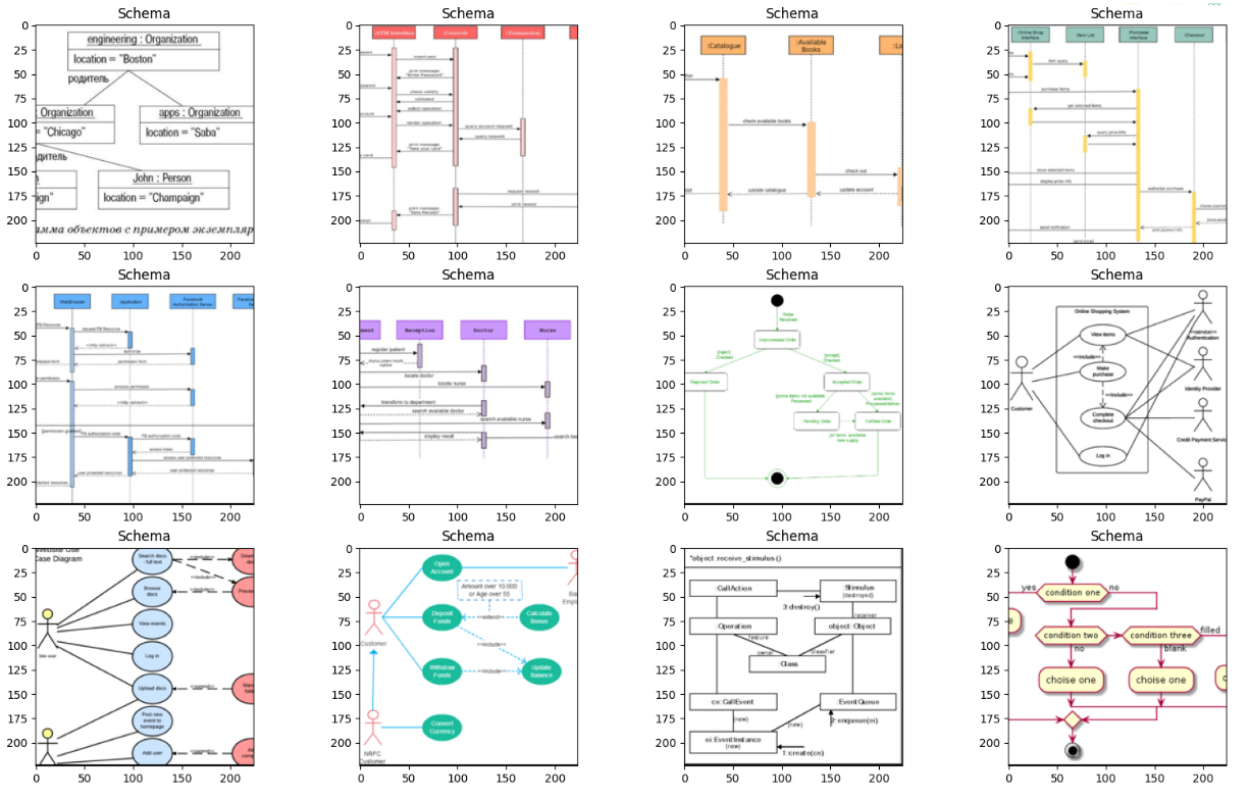


Рисунок 3.2 – Приклади зображень із датасету Schemas

3.2 Навчання нейронної мережі

У роботі було використано мережу MobileNetV2 – це вільно розповсюджувана навченна нейронна мережа, яка вже може виділяти ознаки зображень та приймати на основі цього рішення. Достатньо лише перенавчити шар прийняття остаточного рішення – до якого класу належить зображення.

У даному випадку, відповіді останнього шару були змінені на наступні варіанти: зображення (picture) та схема (schema). Щоб навчити цей шар, достатньо створити відносно невеликий датасет та провести кілька епох навчання. Тож, було створено датасет із 1200 зображеннями, з яких 1100 було відправлено для навчання моделі впродовж 30 епох. Для більшої точності обробки, яскравість зображень була нормована. Зображення із помітками, до якого класу вони мають відноситися, та перемішані були відправлені до мережі задля її навчання. За 8 хвилин (413,9 секунд) мережа завершила своє навчання та показала точність 0,977 при роботі з тестовими даними.

Таке просте й швидке навчання відбулося за рахунок вже навчених середніх шарів мережі, які виділяють ознаки й групують їх. Вони не були перенавчені, адже з таким не великим набором навчальних даних, ваги скоріше б стали гіршими.

3.3 Опис обраних дескрипторів

На основі досліджень, проведених у попередніх роботах, було виявлено, що для задачі виявлення плагіату зображень у текстових файлах найкраще підходять дескриптори SIFT та AKAZE.

Хоча SIFT виявився лише п’ятим за швидкістю обробки дескриптором, його точність у знаходженні відповідностей характерних точок є неймовірно високою. При роботі з однорідним масштабом, він показує відмінні

результати при збільшенні зображень навіть у 10 разів, проте при зменшенні зображення показники трохи гірші, при зменшенні масштабу до 0,2 точність падає у два рази. При неоднорідному масштабуванні, SIFT залишається більш-менш точним при $k \in (0{,}5;\ 2)$. Проте, навіть ці результати є найкращими, що робить його безсумнівним кандидатом для застосування, при проєктуванні прототипу застосунку з пошуку плагіату зображень у текстових документах.

Проте, така слабкість алгоритму, як великий час опрацювання може мати серйозні наслідки для роботи застосунку, тож було вирішено також узяти, для подальшого порівняння співвідношення якості та часу роботи програми, дескриптор AKAZE, через його оптимальні показники швидкодії та точності обчислень. Він має досить гарну точність при роботі з однорідним масштабом у діапазоні $k \in (0{,}5;\ 10)$ та неоднорідному на $k \in (0{,}6;\ 1{,}5)$.

3.4 Розробка прототипу застосунку

3.4.1 Проєктування електронної колекції зображень

Одним з ключових елементів програми є база даних. У ній буде збережено зображення, які будуть еталонними при перевірці на плагіат. Крім того, щоб запобігти чисельним розрахункам, дескриптори зображень та інформацію по їх гістограмам також буде збережено у базі. Для структуризації цих даних, було створено три таблиці, їх зв’язки можна переглянути на рисунку 3.3.

Таблиця imagedto зберігає у собі лише зображення, його тип й файл, в якому його вперше було відмічено й є ключовою таблицею схеми. Descriptorsdto відповідає за збереження обчислених характерних точок зображення, для їх подальшого використання. Pixel_brightnessdto зберігає у собі інформацію по гістограмам зображень й використовується при перевірці першого критерію програми.

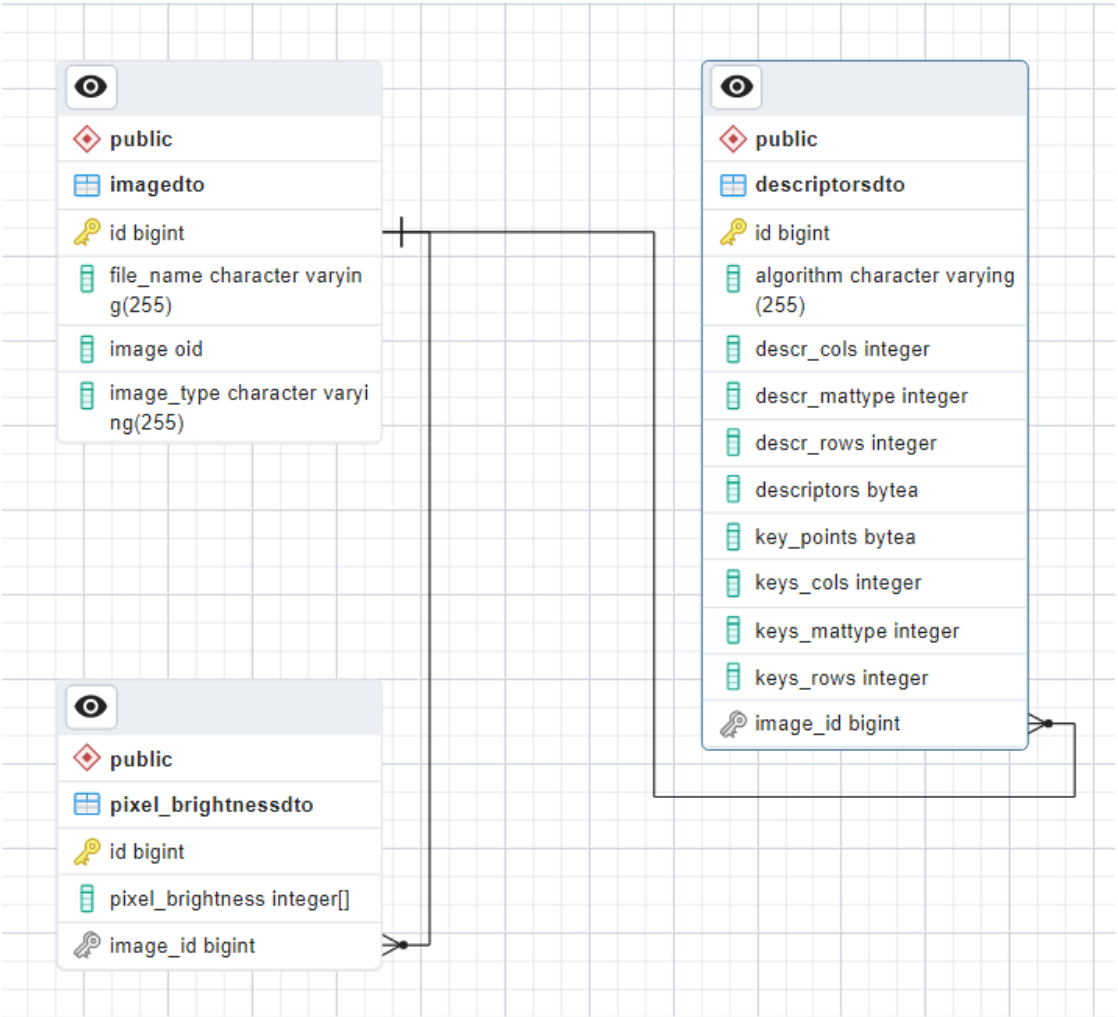


Рисунок 3.3 – Схема бази даних застосунку

3.4.2 Використані програмні засоби та технології

У ході роботи, було розроблено програмний застосунок, який базується на наступному програмному забезпеченні:

- Java 8;
- OpenCV 4.5.1;
- PostreSQL 9.6;
- NodeJS 12.16.0;
- React 17.0.2;
- MobileNetV2;
- Python 3.9.

3.4.3 Ілюстрація роботи застосунку

Прототип програми з перевірки текстових документів на плагіат було розроблено у вигляді вебзастосунку. У ході даної роботи прототип було покращено шляхом додавання кластеризації за допомогою нейронної мережі на два класи – зображення та схеми. Також було змінено алгоритм роботи з гістограмами.

3.4.3.1 Підготовка колекцій тестових документів

Для тестування роботи застосунку, було створено серію текстових документів із зображеннями. Далі наведено приклади, як створювалися три з таких документів. Інші створювалися аналогічно.

Перший документ містить зображення (рис. 3.4), які має бути одразу збережено до бази даних, оскільки вона ще не була наповнена.

Другий файл містить кілька зображень з first.docx, зі зміненими масштабами, та кілька нових зображень (рис. 3.5).

Схожим чином, як і другий, було організовано третій документ, проте у розширені pdf. Він так само містить нові та вже існуючі у базі даних зображення (рис. 3.6, 3.7).

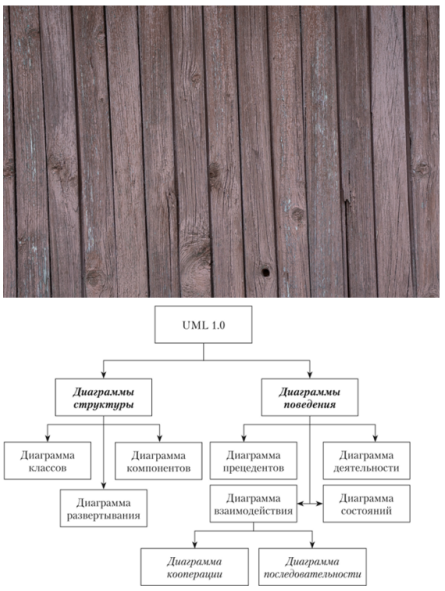
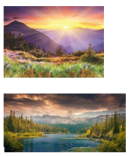


Рисунок 3.4 – Приклади зображень у документі first.docx



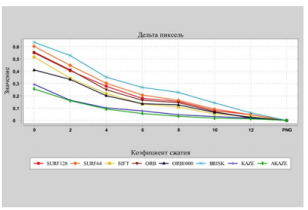
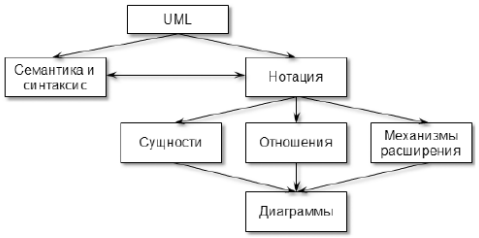


Рисунок 3.5 – Приклад зображень у документі second.docx

Рисунки 3.6 – Приклади зображень у документі third.pdf, частина 1



Рисунки 3.7 – Приклади зображень у документі third.pdf, частина 2

3.4.3.2 Ілюстрація роботи застосунку

Під час тестування застосунку були використані завчасно створені електронні документи. Інтерфейс застосунку складається з двох інтерактивних елементів – поле вибору документу й кнопки запуску обробки запиту (рис. 3.8).

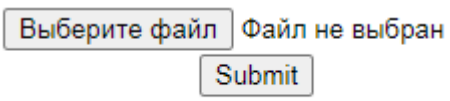
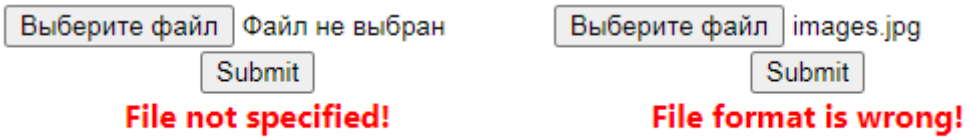


Рисунок 3.8 – Базові елементи інтерфейсу

При спробі виконати запит без введеного файлу, або з файлом з розширенням відмінним від doc чи pdf, буде відображено помилку з відповідним повідомлення. Приклади цих помилок зображено на рисунку 3.9.



а) б)

Рисунок 3.9 – Помилки при обранні файлу:
а) файл не обрано; б) файл має хибне розширення файлу

При завантаженні файлу правильного розширення, він буде відправлений на обробку. Через деякий час, після обробки інформації сервером, має надійти відповідь, у якій буде вказано чи були знайдені зображення, що є підозрілими на плагіат.

Якщо таких зображень не було виявлено, з’явиться повідомлення, про відсутність плагіату у документі, як зображено на рисунку 3.10.

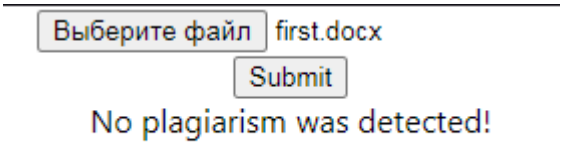


Рисунок 3.10 – Результат обробки файлу без зображень, підозрілих на плагіат

Якщо файл містить плагіатні зображення та їх було знайдено, їх буде виведено на сторінку вебзастосунку. Це буде зроблено у вигляді трьох колонок – зображення з файлу, яке було висунуте як підозріле на плагіат, зображення, які знайшли дескриптори SIFT та AKAZE. На останніх буде

вказано час, який даний алгоритм витратив на пошук. Вивід цієї інформації відображено на рисунку 3.11.

Рисунок 3.11 – Результат обробки файлу із зображенням, підозрілим на плагіат

3.4.3.3 Дослідження швидкодії та точності застосунку

Основною метою цього дослідження було збільшення швидкості роботи застосунку й зменшення кількості операцій з дескрипторами, шляхом кластеризації зображень на фотозображення і схеми й більш точної класифікації за допомогою гістограм.

Щоб проаналізувати, наскільки корисними є попередня обробка зображень за допомогою класифікації та гістограм, було проведено три запуски програми з різними налаштуваннями:

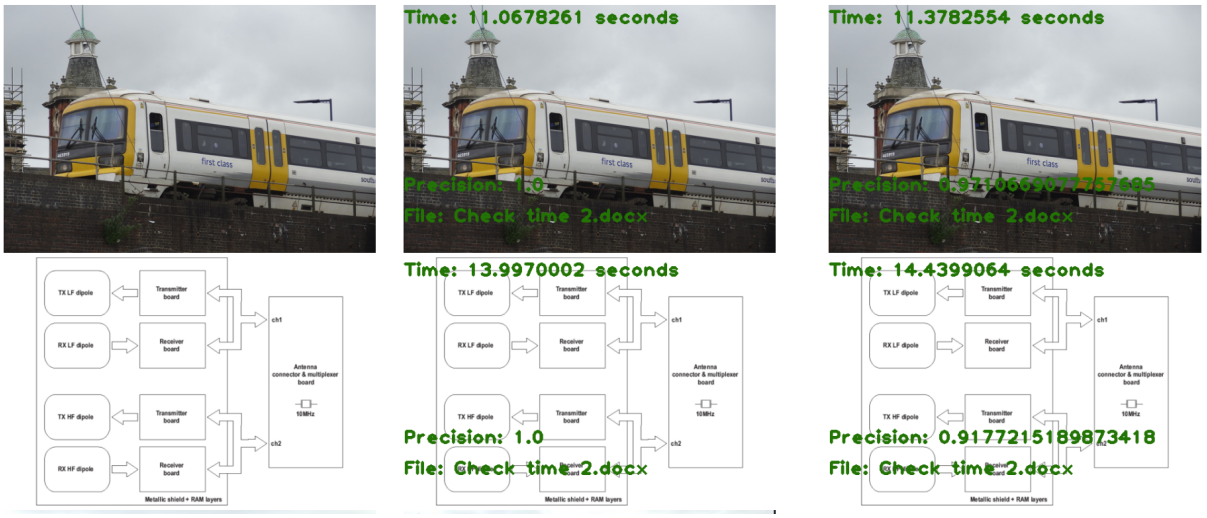
- перевірка лише дескрипторів зображень;
- перевірка спочатку гістограм зображень, потім дескрипторів;
- класифікація зображень, перевірка їх гістограм і після цього дескрипторів.

Під час перевірки, у базі знаходилося біля 120 збережених тестових зображень. Зображення, які були використані для оцінки роботи системи з різними рівнями попередньої обробки, були завчасно збережені у базу. Дані про швидкість роботи (виключно етапу порівняння дескрипторів) та точність оцінки виводилися написом на зображенні.

На рисунках 3.12, 3.13 відображенні результати роботи програми без попереднього відсіювання явно не схожих зображень з бази даних. Через це, для знаходження відповідностей перевіряються усі зображення у базі. І в такому випадку, програма завершить перевірку коли знайде відповідність, що задовольняє порогу критеріїв чи коли перевірить усю базу. В обох випадках час буде збільшуватись разом із збільшенням розмірів бази. В першому вона також буде залежати, від положення відповідного зображення у базі – чим швидше воно надійде на порівняння, тим швидше програма завершить обробку.

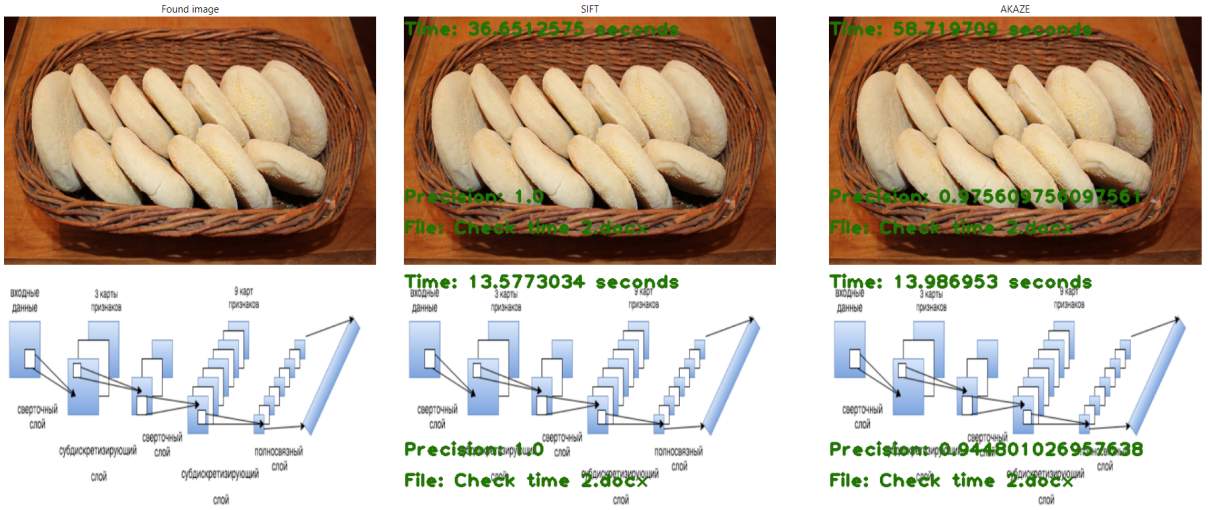
На рисунках 3.14, 3.15 відображено результати роботи застосунку із доданням до алгоритму етапу з перевіркою зображень за допомогою гістограм, як було описано в критерії 1. В даному випадку очікується, що більша частина не схожих зображень буде відсіяна, що зменшить час аналізу дескрипторами.

На рисунках 3.16, 3.17 відображено результати роботи застосунку із доданням до алгоритму етапу з перевіркою зображень за допомогою гістограм та кластеризації. Очікується, що класифікація зменшить кількість зображень, що можуть бути надані на перевірку, навіть до обробки гістограмами.



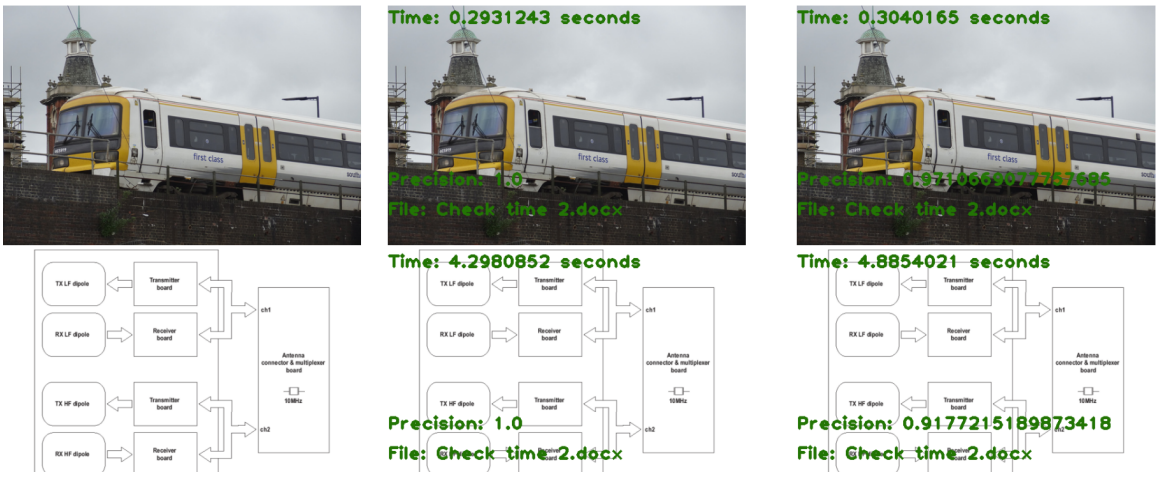
а) б) в)

Рисунок 3.12 – Результат обробки файлу із зображеннями (без попередньої обробки): а) оригінал зображення; б) результат обробки SIFT; в) результат обробки AKAZE



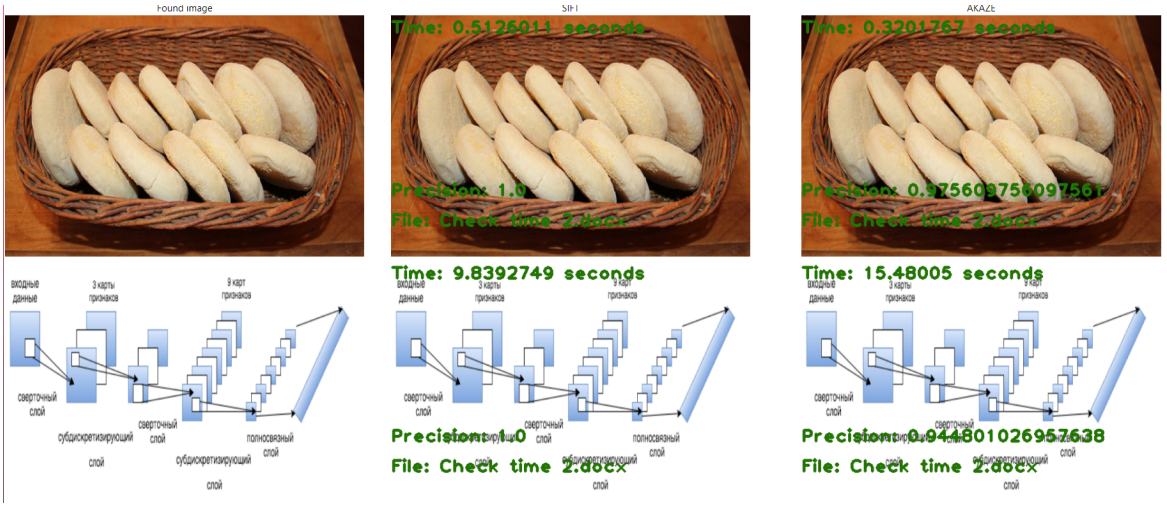
а) б) в)

Рисунок 3.13 – Результат обробки файлу із зображеннями (без попередньої обробки): а) оригінал зображення; б) результат обробки SIFT; в) результат обробки AKAZE



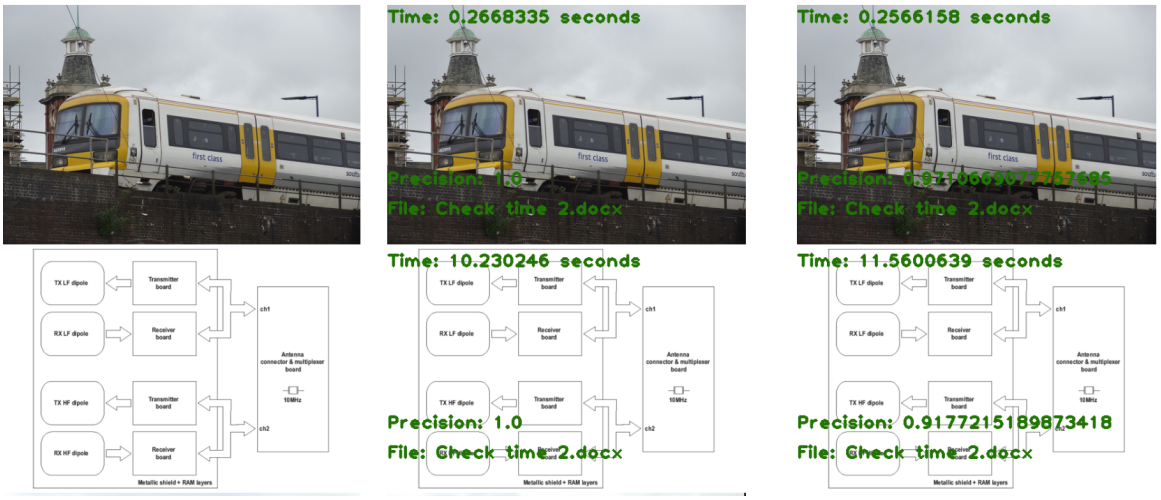
а) б) в)

Рисунок 3.14 – Результат обробки файлу із зображеннями (з попередньою обробкою гістограмами): а) оригінал зображення; б) результат обробки SIFT; в) результат обробки AKAZE



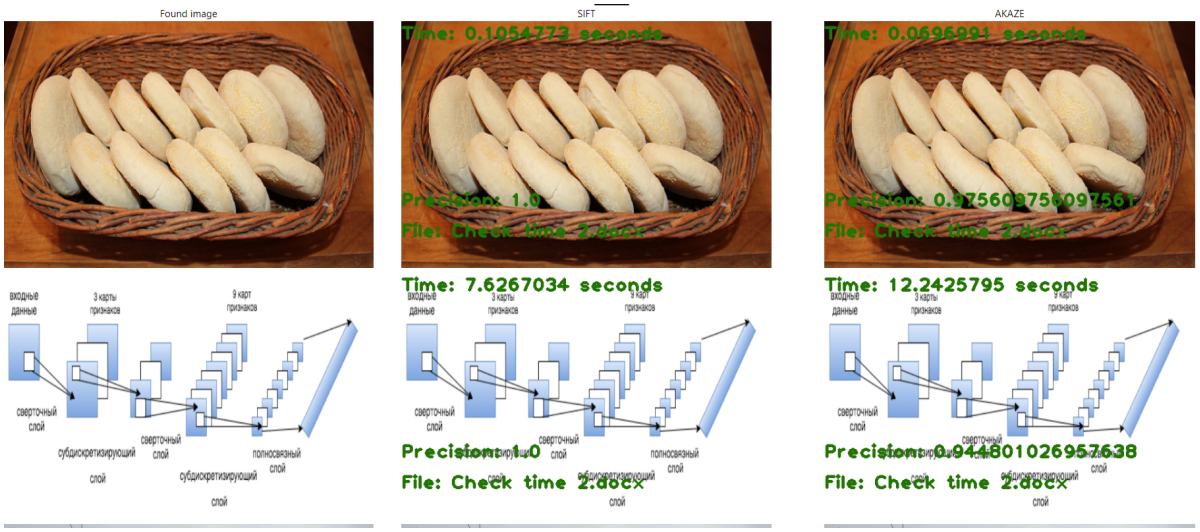
а) б) в)

Рисунок 3.15 – Результат обробки файлу із зображеннями (з попередньою обробкою гістограмами): а) оригінал зображення; б) результат обробки SIFT; в) результат обробки AKAZE



а) б) в)

Рисунок 3.16 – Результат обробки файлу із зображеннями (з попередньою обробкою гістограмами і класифікатором): а) оригінал зображення; б) результат обробки SIFT; в) результат обробки AKAZE



а) б) в)

Рисунок 3.17 – Результат обробки файлу із зображеннями (з попередньою обробкою гістограмами і класифікатором): а) оригінал зображення; б) результат обробки SIFT; в) результат обробки AKAZE

Як видно із цих зображень, у цілому, при роботі програми без будь-якої попередньої обробки, час значно зростав. Додавання хоча б обробки за допомогою гістограм зменшує його у кілька разів. З додаванням етапу класифікації все не так однозначно – у більшості випадків він трохи збільшив швидкість роботи програми (у 2–3 рази в середньому), проте є випадки, коли

швидкість й уповільнювалася. Скоріше за все, це зумовлено складністю для класифікатора обробити це зображення.

Проте це одиничні випадки, усереднені дані кажуть лише про зростання швидкодії роботи програми з додаванням етапів попередньої обробки, навіть на таких, доволі малих розмірах бази даних.

На точність оцінки кожного дескриптору, як видно з прикладів, не вплинули ніякі зміни, що каже про стабільність дескрипторів. Проте у цілому точність роботи SIFT була вищою, близькою до 1, у порівнянні з AKAZE, у якого середня точність 0,9. Крім того, AKAZE зміг знайти плагіат зображень у 99,98% випадках, у той час як SIFT виконував це у всіх 100% випадків.

Більше прикладів роботи програми наведено в додатку А.

ВИСНОВКИ

У рамках кваліфікаційної роботи був розроблений і покращений прототип застосунку з пошуку плагіату зображень у текстових документах. У процесі його вдосконалення були вирішені наступні задачі:

- вивчено питання кластеризації зображень;
- досліджено питання поділу зображень на класи на основі нейромережевого підходу;
- вивчено використання навченних заздалегідь нейронних мереж;
- досліджено використання гістограм у якості ознак зображень для подальшого етапу кластеризації;
- розроблено поетапний класифікатор для аналізу вхідних зображень;
- сформовано датасет для перенавчання нейронної мережі;
- спроєктовано та розроблено застосунок для пошуку плагіату зображень в електронних документах;
- сформовано базу даних зображень для дослідження роботи застосунку;
- проведено дослідження швидкості та якості роботи застосунку.

У ході роботи, найбільше уваги приділялося питанню кластеризації даних, у даному випадку – зображень. Було навчено вихідний шар нейронної мережі, яка вже була навчена виділяти ознаки зображень. Завдяки їй, стало можливим виділяти тип зображення – схема чи фотозображення – що у рази зменшило кількість операцій на етапі порівнянь гістограм та дексрипторів.

Також було змінено алгоритм роботи із гістограмами зображень, для більш точного виявлення схожих файлів, зокрема у випадку із схемами і діаграмами, де більша частина зображення може бути «порожньою». Підхід із розбиттям зображення на зони й побудова гістограми кожної з ним, вирішив питання схожості більшості схем, через велику кількість однотипних точок.

Було проаналізовано роботу застосунку із різними рівнями попередньої обробки зображень й оцінено зміни. Головним висновком є те, що при додаванні цих етапів, швидкість роботи застосунку значно зростає, а отже мета дослідження була досягнута.

Використання попередньої кластеризації дозволяє пришвидшити пошук у 2–3 рази. Точність застосунку є кращою, при використанні дескриптору SIFT, порівнянно з AKAZE, при схожій швидкості обробки.

У подальший дослідженнях було би доцільно збільшити датасет електронних документів, що дозволило б більш обгрунтовано робити висновки щодо точності та швидкодії розробленого методу.

Результати дослідження апробовано у вигляді 3 тез доповідей під час XXXVII Міжнародної науково-практичної конференції «Modern ways of solving the latest problems in science» [19], Восьмої міжнародної науково-технічної конференції «Інформатика, управління та штучний інтелект» [41], XIII-ої МІЖНАРОДНОЇ НАУКОВО-ПРАКТИЧНОЇ КОНФЕРЕНЦІЇ «Free and open source software» [42].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Матеріали XII-ої Ювілейної Міжнародної науково-практичної конференції «Free and Open Source Software», 2020: Попирєв Д.О., Яковлева О.В. (2020). Огляд можливостей бібліотеки OpenCV для аналізу зображень на основі дескрипторів. Матеріали XII-ої Міжнародної науково-практичної конференції «Free and Open Source Software». (17-19 листопада 2020 р.), Харків, С.14, URL: https://foss.kn-it.info/uploads/foss-2020-theses.pdf (дата звернення: 12.11.2022).

2. Попирєв Д.О., Яковлева О.В. (2021). Дослідження інваріантних властивостей дескрипторів для вирішення задачі пошуку плагіату зображень в документах. Матеріали XXV Міжнародного молодіжного форуму «Радіоелектроніка та молодь у XXI столітті», Конференція «Сучасні методи обробки зображень».

3. Яковлева О.В., Попирєв Д.О. (2021). Дослідження проблеми пошуку плагіату зображень у документах та підхід до її вирішення на основі аналізу дескрипторів. Тези сьомої міжнародної науково-технічної конференції "Інформатика, управління та штучний інтелект" (17-19 листопада 2020 року). НТУ «ХПІ», С.78.

4. Гороховатський, В. О., Запорожченко, А. П., Сірик, Т. О., & Тарасенко, О. П. (2020). Дослідження результативності застосування ознак розподілів даних для обчислення релевантності описів зображень.

5. Гороховатський, В. О., Пупченко, Д. В., & Солодченко, К. Г. (2018). Аналіз властивостей, характеристик та результатів застосування новітніх детекторів для визначення особливих точок зображення.

6. Yakovleva, O., & Nikolaieva, K. (2020). Research Of Descriptor Based Image Normalization And Comparative Analysis Of SURF, SIFT, BRISK, ORB, KAZE, AKAZE Descriptors. Advanced Information Systems, 4(4), 89-101.

7. Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. International journal of computer vision, 60(2), 91-110.

8. Bay, H., Tuytelaars, T., & Gool, L. V. (2006, May). Surf: Speeded up robust features. In European conference on computer vision (pp. 404-417). Springer, Berlin, Heidelberg.

9. Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011, November). ORB: An efficient alternative to SIFT or SURF. In 2011 International conference on computer vision (pp. 2564-2571). Ieee.

10. Leutenegger, S., Chli, M., & Siegwart, R. Y. (2011, November). BRISK: Binary robust invariant scalable keypoints. In 2011 International conference on computer vision (pp. 2548-2555). Ieee.

11. Alcantarilla, P. F., Bartoli, A., & Davison, A. J. (2012, October). KAZE features. In European conference on computer vision (pp. 214-227). Springer, Berlin, Heidelberg.

12. Toğaçar, M., Cömert, Z., & Ergen, B. (2021). Intelligent skin cancer detection applying autoencoder, MobileNetV2 and spiking neural networks. Chaos, Solitons & Fractals, 144, 110714.

13. Toğaçar, M., Cömert, Z., & Ergen, B. (2021). Intelligent skin cancer detection applying autoencoder, MobileNetV2 and spiking neural networks. Chaos, Solitons & Fractals, 144, 110714.

14. Daradkeh, Y.I., Gorokhovatskyi, V., Tvoroshenko, I., Gadetska, S., and Al-Dhaifallah, M. (2021) Methods of Classification of Images on the Basis of the Values of Statistical Distributions for the Composition of Structural Description Components, *IEEE Access*, 9, pp. 92964-92973.

15. Gorokhovatskyi, V.O., Tvoroshenko, I.S., and Peredrii O.O. (2020) Image classification method modification based on model of logic processing of bit description weights vector, *Telecommunications and Radio Engineering*, 79(1), pp. 59-69.

16. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., and Zeghid M. (2022) Cluster representation of the structural description of images for effective classification, *Computers, Materials & Continua*, 73(3), pp. 6069–6084.

17. Гороховатский, В. А., Ересько, Ю. Н., Путятин, Е. П., & Стрельченко, В. И. (1990). Локализация объектов на изображениях визуальных сцен. *Автометрия*, 6, 3-7.

18. Путятин, Е. П., Яковлева, Е. В., & Любченко, В. А. (1999). Исследование инвариантных прямых и их применение в алгоритмах нормализации изображений.

19. Матеріали XXXVII Міжнародна науково-практична конференція «Modern ways of solving the latest problems in science», 2022: Попирєв Д.О., Яковлева О.В. (2020). Розробка та дослідження методу виявлення в електронних документах підозрілих на плагіат зображень. XXXVII Міжнародна науково-практична конференція «Modern ways of solving the latest problems in science». (20-23 вересня 2022 р.), Варна, Болгарія, С.470, URL: https://isg-konf.com/uk/modern-ways-of-solving-the-latest-problems-in-science/ (дата звернення: 10.11.2022).

20. Лящинський, П. Б. (2018). Модуль розпаралелення алгоритмів навчання згорткових нейронних мереж.

21. Ayob, A. F., Khairuddin, K., Mustafah, Y. M., Salisa, A. R., & Kadir, K. (2021). Analysis of pruned neural networks (MobileNetV2-YOLO v2) for underwater object detection. In Proceedings of the 11th National Technical Seminar on Unmanned System Technology 2019 (pp. 87-98). Springer, Singapore.

22. Alcantarilla, P. F., & Solutions, T. (2011). Fast explicit diffusion for accelerated features in nonlinear scale spaces. IEEE Trans. Patt. Anal. Mach. Intell, 34(7), 1281-1298.

23. А.Р. Ковтуненко, О.В. Яковлева, В.А. Любченко, & О.В. Янголенко (2020) Дослідження сумісного використання математичної морфології та згорткових нейронних мереж для вирішення задачі

розпізнавання цінників. Вісник Національного технічного університету ХПІ (3). 24-31.

24. Гороховатский, В. А. (2014). Структурный анализ и интеллектуальная обработка данных в компьютерном зрении.

25. Gorokhovatskyi, V., Vasylchenko, A., Manko, K., & Ponomarenko, R. (2018). Дослідження модифікацій методу встановлення релевантності зображень об'єктів за описами у вигляді множини дескрипторів ключових точок. *Системи управління, навігації та зв'язку. Збірник наукових праць*, *5*(51), 74-78.

26. Гороховатский, В. А., Кацалап, С. Ф., & Путятин, Е. П. (1986). Анализ изображений в условиях локальных искажений. *Автометрия*, *6*, 46.

27. Гороховатський, В. О., & Солодченко, К. Г. (2018). Застосування апарату аналізу та оброблення бітових даних у методах класифікації зображень за множиною ключових точок. *Системи управління, навігації та зв'язку*, (2), 63-67.

28. Gorokhovatsky, V. O., Pupchenko, D. V., & Solodchenko, K. G. (2018). Аналіз властивостей, характеристик та результатів застосування новітніх детекторів для визначення особливих точок зображення. *Системи управління, навігації та зв'язку. Збірник наукових праць*, *1*(47), 93-98.

29. Патин, М. В., & Коробов, Д. В. (2016). Сравнительный анализ методов поиска особых точек и дескрипторов при группировке изображений по схожему содержанию. *Молодой ученый*, (11), 214-221.

30. Гороховатський, В. О., Гадецька, С. В., Стяглик, Н. І., & Власенко, Н. В. (2020). Класифікація зображень на підставі ансамблю статистичних розподілів за класами еталонів для компонентів структурного опису.

31. Гадецька, С. В., Гороховатський, В. О., & Стяглик, Н. І. (2020). Вивчення критеріїв інформативності даних при впровадженні апарату дерев рішень у методах структурної класифікації зображень.

32. Zhang, Y., Li, C., Cao, C., & Gao, Y. (2018, December). An Improved ORB Feature Point Matching Algorithm. In Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence (pp. 207-211).

33. Noble, F. K. (2016, November). Comparison of OpenCV's feature detectors and feature matchers. In 2016 23rd International Conference on Mechatronics and Machine Vision in Practice (M2VIP) (pp. 1-6). IEEE.

34. Zalesky, B. A. (2017). DETECTORS OF EXTREMAL KEY POINTS ON IMAGES. *Doklady of the National Academy of Sciences of Belarus*, *61*(5), 37-41.

35. Лютенко, И.В., Чередниченко, О.Ю., Яковлева, Е.В., & Максименко, Е.М. (2015). Models of representation of multi-features objects based on sequential aggregation. *Bulletin of NTU" KhPI". Series: Strategic Management, Portfolio, Program and Project Management*, *3*(1 (1110)), 149-154.

36. Кобилін, О. А., & Творошенко, І. С. (2021). Методи цифрової обробки зображень: навч. посібник. *Харків: ХНУРЕ*.

37. Гороховатський, В. О., & Солодченко, К. Г. (2018). Застосування апарату аналізу та оброблення бітових даних у методах класифікації зображень за множиною ключових точок. *Системи управління, навігації та зв'язку*, (2), 63-67.

38. Путятін, Є. П., Гороховатський, В. О., & Матат, О. О. (2006). Методи та алгоритми комп'ютерного зору: навч. посібник.

39. Ніколаєва, К. Г. (2019). Розробка та дослідження методу нормалізації геометричних перетворень зображень на основі аналізу характерних точок.

40. Гороховатський В.О., Творошенко І.С., Чмутов Ю.В. (2022) Застосування систем ортогональних функцій для формування простору ознак у методах класифікації зображень, *Сучасні інформаційні системи*, 6(3), С. 5–12.

41. Яковлева О. В., Попирєв Д. О. (2021) Дослідження питання пошуку зображень на основі дескрипторів для виявлення плагіату зображень у текстових файлах, *Інформатика, управління та штучний інтелект*, С. 154.

42. Яковлева О. В., Попирєв Д. О. (2021) Огляд продукції компанії-розробника JETBRAINS, *Free and open source software*, С. 25.