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 перший (бакалаврський)

ПРОГРАМНЕ МОДЕЛЮВАННЯ КЛАСИФІКАТОРА ЗОБРАЖЕНЬ НА БАЗІ КРИТЕРІЇВ УНІКАЛЬНОСТІ ТА ІНФОРМАТИВНОСТІ ДЛЯ СТРУКТУРНИХ ОЗНАК
(тема)

Виконав:
студент 4 курсу, групи ІТІНФ-19-1

Шевляков А.С.
(прізвище, ініціали)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник проф. Гороховатський В.О.
(посада, прізвище, ініціали)

Допускається до захисту

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

2023 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти перший (бакалаврський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Шевлякову Артему Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Програмне моделювання класифікатора зображень на базі критеріїв унікальності та інформативності для структурних ознак

затверджена наказом університету від 15 травня 2023 року № 474 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 29 травня 2023 р.

3. Вихідні дані до роботи науково-методична та науково-технічна література з комп'ютерного зору, матеріали конференцій, дані інтернет-мережі, бібліотека комп'ютерного зору OpenCV, середовище розробки Visual Studio Code, мова програмування C#, NuGet package Emgu.CV.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд методів ознакового опису зображень.
2. Реалізація алгоритму класифікації зображень, який використовуватиме критерії унікальності та інформативності.
3. Створення програмного коду для формування редукованого опису і класифікації у базі зображень із застосуванням методу BRISK.
4.Тестування роботи програми з використанням кольорових зображень NFT-персонажів розміром 420×420 пікселів.
5. Оцінка ефективності розробленого методу класифікації зображень.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність проблеми класифікації зображень, постановка задачі, математичні моделі, тестові зображення, результати експериментів.

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	10.04.2023	
2	Аналіз завдання, підбір літератури	11.04.23-15.04.23	
3	Аналіз літератури з досліджуваної проблеми	16.04.23-19.04.23	
4	Аналіз методів опису зображень	20.04.23-27.04.23	
5	Розробка алгоритму класифікації зображень, на базі критерії унікальності та інформативності.	28.04.23-12.05.23	
6	Програмна реалізація	13.05.23-21.05.23	
7	Оформлення пояснювальної записки	22.05.23-26.05.23	
8	Перевірка на плагіат	27.05.23	
9	Рецензування	28.05.23	
10	Підготовка презентації та доповіді	29.05.23-30.05.23	
11	Занесення роботи в електронний архів	31.05.23	
12	Попередній захист кваліфікаційної роботи	07.06.23	

Дата видачі завдання 10 квітня 2023 р.

Студент _____
(підпис)

Керівник роботи _____ __ проф. Гороховатський В.О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 66 с., 3 табл., 16 рис., 46 джерел.

КЛЮЧОВА ТОЧКА, ДЕСКРИПТОР, ДЕТЕКТОР BRISK, ГЕММІНГОВА ВІДСТАНЬ, ІНФОРМАТИВНІСТЬ.

Об’єтом роботи є методи класифікації зображень з відбором за критеріями унікальності та інформативності структурних ознак.

Метою роботи є створення класифікатора для вирішення проблеми швидкого віднесення зображень до класів та їх розпізнавання у системах комп’ютерного зору. За допомогою відбору частини дескрипорів за показниками інформативності та унікальності класифікатор забезпечує швидку та точну класифікацію зображень у реальному часі.

Застосовано методи класифікації зображень на підставі опису як множини дескрипторів ключових точок. Відстань Геммінга була використана при порівнянні двох бінарних рядків, а саме бінарних дескриптори ключових точок при зіставленні зображень.

Проведено програмне моделювання класифікатора за результатом відбору підмножини інформативних ознак. Результати моделювання на навчальній множині описів зображень підтверджують ефективність розроблених методів класифікації та забезпечують достатній рівень точності.

KEYPOINT, DESCRIPTOR, DETECTOR BRISK, HEMMING DISTANCE, INFORMATIVENESS

The object of the work is image classification methods with selection based on the criteria of uniqueness and informativeness of structural features.

The aim of the work is to create a classifier to solve the problem of quickly assigning images to classes and their recognition in computer vision systems. Using the selection of a part of the descriptors according to the indicators of informativeness and uniqueness, the classifier provides fast and accurate classification of images in real time.

Image classification methods based on the description of a set of descriptors of key points are applied. The Hamming distance was used when comparing two binary strings, namely binary descriptors of key points when matching images.

Software modeling of the classifier based on the selection of a subset of informative features was carried out. The simulation results on the training set of image descriptions confirm the effectiveness of the developed classification methods and provide a sufficient level of accuracy.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 6
Вступ 7
1 Аналіз структурних методів розпізнавання зображень 9
1.1 Визначення і основні властивості ключових ознак зображення ... 10
1.2 Методи формування ключових точок 12
1.3 Способи зіставлення структурних описів 18
1.4 Постановка задачі 22
2 Класифікація зображень з використанням критеріїв унікальності та інформативності 23
2.1 Огляд сучасних моделей класифікації 23
2.2 Застосування параметрів унікальності та інформативності 34
2.3 Застосування голосування у структурних методах 40
3 Програмне моделювання класифікатора зображень 43
3.1 Обґрунтування вибору середовища програмної реалізації 43
3.2 Особливості програмної реалізації 50
3.3 Інструкція користувача 57
3.4 Аналіз експериментальних результатів 58
Висновки 61
Перелік джерел посилання 62

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

КЗ – комп’ютерний зір

КТ – ключові точки

ХО – характерні ознаки

МВ – манхеттенська відстань

МХ – метрика Хемінга

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

SURF – Speeded Up Robust Features (прискорені стійкі ознаки)

FAST – Features from Accelerated Segment Test (ознаки тестування прискореного сегмента)

BRIEF – Binary Robust Independent Elementary Features (короткі двійкові надійні незалежні елементарні ознаки)

BRISK – Binary Robust Invariant Scalable Keypoints (двійкові надійні інваріантні масштабовані ключові точки)

ORB – Oriented FAST and Rotated BRIEF (орієнтований FAST і обернутий BRIEF)

VSC – Visual Studio Code

Open CV – Open Source Computer Vision Library (бібліотека комп’ютерного зору з відкритим кодом)

НМ – нейронна мережа

ПЗ – порівняння із зразком

НМ – нейронні мережі

СМ – статистичні методи

ССМ – структурні та синтаксичні методи

АДП – алгоритм динамічного програмування

СР – синтаксичне розпізнавання

КВГ – контекстно-вільна граматика

МОВ – метод опорних векторів

ВСТУП

Комп’ютерний зір (КЗ) – галузь штучного інтелекту, яка займається створенням алгоритмів та програм для оброблення, аналізу та інтерпретації зображень та відео. Головна мета КЗ полягає в тому, щоб навчити комп’ютери розпізнавати та розуміти інформацію, що міститься у зображеннях [1-12].

КЗ вирішує велику кількість завдань, що включають розпізнавання об’єктів і класифікацію, виявлення об’єктів, розпізнавання осіб, трекінг об’єктів, оптичне розпізнавання символів та ін. Ці завдання виконуються за допомогою розробки алгоритмів та програм для обробки, аналізу та розуміння зображень та відео, щоб комп’ютери могли розуміти інформацію, яка може бути вилучена із зображень.

Застосування КЗ включають діагностику та лікування захворювань, автоматичну навігацію та управління роботами, контроль якості продукції, забезпечення безпеки та ін [1, 13-15].

Обробка зображень включає використання алгоритмів і технік для поліпшення якості зображення. Межі між обробкою зображень, аналізом зображень та КЗ є відносними і можуть змінюватись в залежності від контексту. Загалом, обробка зображень фокусується на перетворенні зображень у цифровий формат та застосуванні до них операцій з поліпшення якості, видалення шуму, зміни контрасту, коригування кольорів і т.д. Аналіз зображень займається вилученням інформації із зображень та її інтерпретацією, наприклад розпізнаванням об’єктів, виявленням певних ознак або особливостей. КЗ включає розробку алгоритмів і програм, які дозволяють комп’ютерам бачити і розуміти зображення і відео, наприклад, розпізнавання образів, визначення об’єктів, трекінг руху і т.д. Хоча ці області тісно пов’язані між собою та перекриваються, вони мають свої унікальні завдання та методи вирішення.

Загалом, аналіз та розпізнавання зображень мають широкий спектр застосувань та відіграють важливу роль у багатьох сферах життя,

допомагаючи вирішувати різноманітні завдання, збільшувати ефективність та підвищувати якість життя [2, 16-18].

Класифікація зображень є автоматичним процесом визначення класу або категорії, до якої відноситься зображення, ґрунтуючись на його властивостях і змісті.

Метод класифікації зображень на основі критеріїв унікальності та інформативності для структурних ознак використовує текстури, форми та кольори, щоб автоматично визначити клас чи категорію, до якої належить зображення. Унікальність кожного класу чи компоненти опису гарантується наявністю унікальних структурних ознак, які відрізняють його від інших класів та забезпечують більш точну класифікацію. При виборі структурних ознак важлива їхня інформативність, тобто показник класифікаційної значущості для компоненти опису, який може бути використано для класифікації, щоб зменшити кількість помилок у класифікації [1, 3].

1 АНАЛІЗ СТРУКТУРНИХ МЕТОДІВ РОЗПІЗНАВАННЯ ЗОБРАЖЕНЬ

Існуючі методи розпізнавання зображень стикаються з проблемами, пов’язаними зі складністю розпізнавання об’єктів на зображеннях у реальних умовах, нестачею якісних даних для навчання алгоритмів, можливістю помилкових спрацьовувань чи перепусток та складністю адаптації алгоритмів для різних пристроїв. Також проблемами можуть виступати різноманітні візуальні спотворення через геометричні перетворення та/або інші зовнішні перешкоди [19–24].

В останні роки відбулося багато нових змін у розвитку систем розпізнавання зображень. Однією з найважливіших обставин було поява глибокого навчання і нейронних мереж (НМ), які значно підвищили точність розпізнавання і дозволили обробляти великі обсяги даних [1, 25–29].

Також відбувся значний прогрес у галузі обробки природної мови, що дозволяє системам розпізнавання зображень більш точно та ефективно інтерпретувати текстову інформацію, пов’язану із зображеннями.

Системи розпізнавання зображень стали більш широко використовуваними у різних галузях, включаючи медицину, виробництво, безпеку та багато інших. Вони використовуються для розпізнавання осіб, визначення дефектів на виробничих лініях, аналізу медичних зображень та багатьох інших завдань.

Також з’явилися нові методи, такі як генеративні адверсаріальні мережі, які можуть бути використані для створення реалістичних зображень, а також мережі, здатні адаптуватися до змін у зображеннях в реальному часі.

Існують і інші технології та тенденції, такі як автоматичне доповнення зображень, підвищення роздільної здатності зображень, розпізнавання геометричних форм та застосування систем розпізнавання зображень у поєднанні з іншими технологіями, наприклад, доповненої реальності та віртуальної реальності [30–35].

1.1 Визначення і основні властивості ключових ознак зображення

Люди зазвичай легко вирішують завдання розпізнавання підсвідомо, а створення ж систем, які б реалізовували цю задачу так ефективно, як наш мозок надзвичайно складна задача . Основна проблема полягає в тому, що часто буває важко визначити ознаки, на основі яких можна проводити розпізнавання. Однак для деяких завдань, де такі ознаки можна виділити, штучні системи розпізнавання широко застосовуються.

Характерні ознаки (ХО) – це вектори, які відповідають координатам ключових точок на зображенні та мають властивість інваріантності до геометричних перетворень об’єктів у полі зору. Деякі ознаки є природними, оскільки їх можна визначити візуальним аналізом зображення, тоді як інші, які називаються штучними ознаками, отримуються в результаті спеціальної обробки або вимірювань. Природні ознаки включають яскравість та текстуру різних областей зображення, а також форму контурів об’єктів [35–40].

Ключова точка (КТ) – точка, що знаходиться у місці значної зміни інтенсивності зображення.

Детектор – це метод або алгоритм, який перетворює об’єкт (наприклад, зображення, аудіофайл або текст) у чисельний опис, що дозволяє використовувати його для розпізнавання або порівняння з іншими об’єктами. Детектори широко застосовуються в комп’ютерному зорі, машинному навчанні та обробці сигналів. Вони можуть бути основою для створення моделей машинного навчання, таких як НМ, які використовуються для класифікації або розпізнавання образів [2, 4, 8, 41].

У процесі аналізу зображень ключові точки вилучаються із зображення за допомогою алгоритмів детектування, таких як SIFT, SURF, BRISK, ORB та ін. Далі отримані дескриптори використовуються для зіставлення та порівняння ключових точок на різних зображеннях [4–8].

Ключові точки повинні мати такі основні характеристики:

– локалізація: точне визначення положення КТ на зображенні для можливості використання їх у розпізнаванні та класифікації;

– унікальність: кожна КТ повинна мати унікальні характеристики, що відрізняють їх від інших точок на зображенні;

– інваріантність: ключові точки повинні бути стійкими до деяких перетворень зображення, таких як поворот, масштабування або зміна освітлення;

– висока роздільна здатність: ключові точки повинні мати високу роздільну здатність, щоб забезпечити точність і надійність у подальшому використанні.

Формування КТ зображення – це процес вибору певних точок на зображенні для подальшого аналізу та зіставлення з КТ інших зображень.

Процес формування множини дескрипторів КТ включає кілька етапів:

– виявлення ключових точок на зображенні: алгоритми вибирають КТ, які є яскраво вираженими чи унікальними на зображенні;

– локалізація КТ: визначаються координати кожного обраного КТ на зображенні;

– обчислення дескрипторів для кожного КТ: для кожного КТ обчислюється числовий вектор, що описує характеристики точки точки на зображенні;

– фільтрування дескрипторів: деякі дескриптори можуть бути відфільтровані, якщо вони не відповідають певним критеріям якості;

– нормалізація дескрипторів: дескриптори можуть бути нормалізовані, щоб забезпечити їх інваріантність до масштабування, поворотів та спотворень зображення;

– зіставлення дескрипторів: дескриптори використовуються для зіставлення КТ на різних зображеннях, що дозволяє визначати відповідність між об’єктами на зображеннях або їх змінами в часі [9–11].

Етапи можуть відрізнятися залежно від конкретного способу побудови дескрипторів КТ. Перехід від зображення до множини ХО дозволяє значно

зменшити обсяг інформації, зберігаючи при цьому точність та захищеність від перешкод. У більшості випадків ХО представлені числовими векторами фіксованої розмірності з асоційованими значеннями:

$$\lambda^k = (\lambda_1, \ldots, \lambda_n)^k, \tag{1.1}$$

$$\lambda_i \in R^1, i = \overline{1, n}. \tag{1.2}$$

1.2 Методи формування ключових точок

Методи формування КТ можуть змінюватись в залежності від задачі та використовуваного алгоритму. Нижче розглянемо найпоширеніші з них.

Scale-Invariant Feature Transform (SIFT) і вперше був представлений у 2004 році Д. Лоу, Університет Британської Колумбії. SIFT – це незмінність до масштабу та обертання зображення. Цей алгоритм запатентований, тому його включено до модуля Non-free в OpenCV [12–14]. Алгоритм включає такі етапи:

– перетворення масштабу: зображення масштабується на різні рівні, щоб забезпечити інваріантність масштабу зображення;

– виділення КТ: на кожному рівні масштабування виділяються точки з високою локальною мінливістю інтенсивності;

– обчислення орієнтації: кожної ключової точки визначається її орієнтація шляхом побудови гістограми напрямів градієнтів в околиці точки;

– локальний опис: для кожної ключової точки будується локальний опис її околиці у вигляді векторного дескриптора, який містить інформацію про направлення градієнтів у околах;

– відбір ключових точок: із усіх ключових точок відбираються найбільш стійкі до різних трансформацій, використовуючи поріг стабільності;

– матчінг: ключові точки порівнюються між кількома зображеннями для пошуку відповідностей і вирішення різних завдань КЗ.

Метод SURF використовується для пошуку об’єктів на зображенні, але він працює не з об’єктами (не виділяє об’єкт з фону). Метод розглядає зображення як єдине ціле та шукає особливості цього зображення. При цьому особливості можуть бути всередині об’єкту, на фоні, на точках межі об’єкту та фону. SURF не знайде КТ усередині таких об’єктів простої форми та текстури. КТ будуть знайдені на межі об’єкту з фоном або лише на фоні.

Метод SURF формує $O \subset B_1^n$ (кінцеву мультимножину) із підмножини $B_1^n \subset B^n$ векторів $o = (o_1, \dots, o_n),\ o \in O,\ n = 64 \vee 128$. Підмножина B_1^n визначена як множина n -мірних дійсних векторів, евклідова норма яких дорівнює одиниці [1, 7, 14]:

$$\|o\| = \sqrt{\sum_{i=1}^{n} o_i^2} = 1. \tag{1.3}$$

На практиці ця умова реалізується у наближеному вигляді:

$$B_1^n = \{o | o \in B^n,\ \|o\| \approx 1\}. \tag{1.4}$$

Опис SURF може містити сотні векторів, що суттєво уповільнює обробку.

Завдання скорочення числа векторів передбачає побудову стисненого опису O^* з урахуванням відображення $\Omega: O \to O^*$.

Варіантом опису є формування підмножини $O^* \subset O$ значно меншої потужності шляхом застосування процедури відбору (редукції) ознак O. «Редукція множини ознак» є способом компресії даних з метою зниження витрат на розпізнавання. У теорії розпізнавання цей процес називають формуванням підмножини «значимих», «унікальних» чи «інформативних» ознак.

Алгоритм методу FAST для швидкого виявлення ключових точок у зображеннях ґрунтується на порівнянні яскравості пікселів у заданому радіусі. Він працює наступним чином: спочатку вибирається піксель і задається поріг яскравості. Потім визначається, чи є обраний піксель яскравішим або темнішим за порогове значення. Далі вибирається радіус R і перевіряється, чи є *R* послідовних пікселів, яскравість яких відрізняється від яскравості обраного пікселя. Якщо такі пікселі є, то вибраний піксель вважається ключовою точкою. Цей процес повторюється для кожного пікселя у зображенні [15].

Шляхом порівняння яскравості пікселів усередині кола з яскравістю центрального пікселя *C* можна отримати один із трьох можливих результатів: піксель може бути яскравішим, темнішим або мати схожу яскравість із центральним пікселем *C* (рис. 1.1):

$$\begin{aligned} I_p &> I_C + t \\ I_p &< I_c - t \\ I_C - t &< I_p < I_C + t, \end{aligned} \tag{1.5}$$

де I – яскравість пікселів;

t – деякий заздалегідь фіксований поріг яскравості.

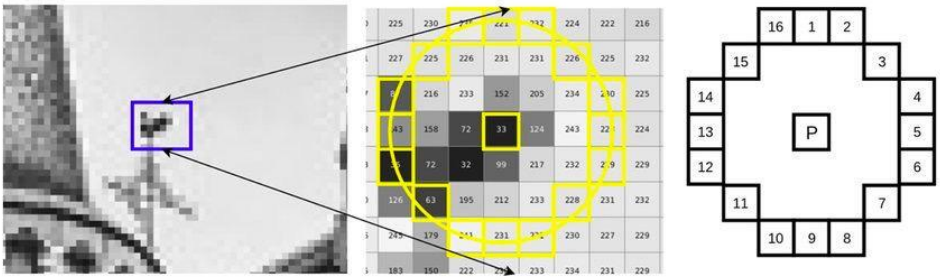

Рисунок 1.1 – Розглядувана околиця точки p детектору FAST

Після виявлення КТ створюються дескриптори кожної з них. Для створення дескриптора визначається невелика область довкола КТ, яка називається патчем (рис. 1.2). Патч – це квадрат із певними розмірами та

пікселями навколо ключової точки. Для представлення об’єкта як патчів використовуються бінарні вектори ознак, які створюються з урахуванням попарного порівняння яскравостей пікселів у патчі. Ці вектори складаються лише з двійкових значень (1 або 0) та називаються дескрипторами бінарних ознак. Кожна КТ має свій дескриптор, який є рядком з 128-512 бітів.

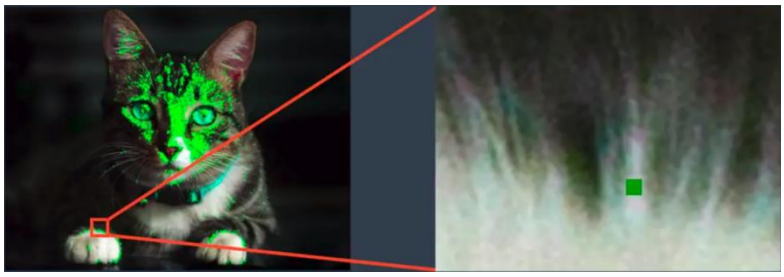

Рисунок 1.2 – Зелений квадрат у патчі є ключевою точкою

BRIEF обробляє зображення на рівні пікселів, тому воно дуже чутливе до шуму. Шляхом попереднього згладжування патча цю чутливість можна зменшити, тим самим підвищивши стабільність та відтворюваність дескрипторів. З тієї ж причини зображення мають бути згладжені, перш ніж їх можна буде осмислено диференціювати під час пошуку країв.

Для згладжування зображення у методі BRIEF застосовується ядро Гауса. Чим складніше порівняння, тим важливіше правильно вибрати дисперсію ядра Гауса, щоб досягти хорошої продуктивності [16].

Маючи згладжений патч зображення p, треба створити двійковий вектор ознак із цього патча. Далі створюється бінарний вектор ознак бінарних тестових (τ) відповідей. Двійковий тест τ визначається так:

$$\tau(p;x,y) = \begin{cases} 1, p(x) < p(y) \\ 0, p(x) \geq p(y) \end{cases}, \quad (1.6)$$

де $p\ (x)$ – інтенсивність p у точці x;

$p\ (y)$ – інтенсивність p у точці y.

Вибір набору з n(x,y)-пар розташування однозначно визначає набір бінарних тестів. Де n – це довжина вектора бінарних ознак, і це може бути 128, 256 і 512 [12].

BRISK алгоритм виявлення ключових точок і вичислення їх дескрипторів, який базується на комбінації детектора FAST і дескриптора BRIEF. BRISK був розроблений для досягнення високої продуктивності та масштабованості, а також для забезпечення інваріантності при зміні масштабу та повороту зображення [1, 17, 18].

Як і в BRIEF, у BRISK використовується бінарний дескриптор, але на відміну від BRIEF, який використовує випадкові патчі, BRISK будує дескриптор на основі локальних бінарних шаблонів (LBP) патчів зображення.

У BRISK знаходження максимумів відбувається не тільки на оригінальному зображенні, але і в масштабованому просторі. Такий простір складається з n октав c_i і n внутрішніх октав d_i, $i = \{0,1, ..., n-1\}$. Зазвичай n дорівнює 4. Октави складаються шляхом зменшення масштабу вихідного зображення в 2 рази (рис. 1.3). Кожна внутрішня октава d_i розташовується між октавами c_i і c_{i+1}.

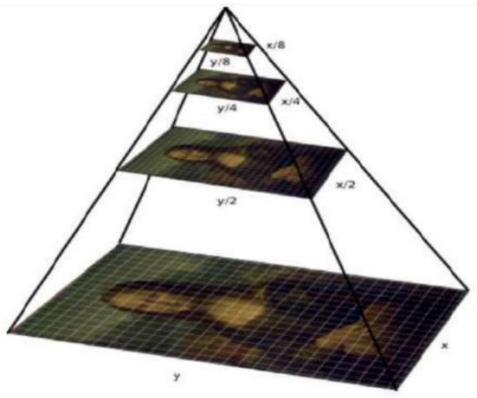

Рисунок 1.3 – Приклад піраміди зображення розбиттям на октави

Метод BRISK має властивість інваріантності до поворотів, а також частково змін масштабу і яскравості зображення. При цьому реалізація методу досить швидка і не залежить від роздільної здатності зображення.

Дескриптори функцій BRISK представляються як бінарних векторів ознак з розміром, кратним ступеня двійки, що прискорює процес зіставлення множини дескрипторів (рис. 1.4).

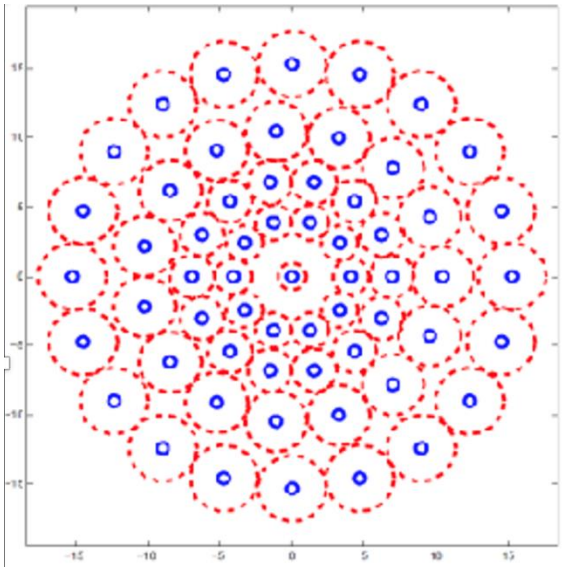

Рисунок 1.4 – Зразок вибірки BRISK

На рисунку 1.4 зображено область, що оточує ключову точку, вона розбивається на 60 ділянок. Вибірка складається з концентричних кілець.

Метод ORB, який є злиттям двох інших методів – FAST (Features from Accelerated Segment Test) та BRIEF (Binary Robust Independent Elementary Features), має багато переваг. Він забезпечує інваріантність до поворотів, масштабу та освітлення зображення, зберігаючи переваги обох методів. Для виявлення ключових точок на зображенні ORB використовує метод FAST, а для створення дескрипторів на основі цих ключових точок, застосовує алгоритм BRIEF. Для створення орієнтованих дескрипторів ORB використовує множину напрямків, що робить його більш стійким до поворотів зображення. Крім того, він застосовує механізми для скорочення кількості ключових точок, що робить його ефективнішим у порівнянні з іншими методами. Загалом ORB забезпечує швидку обробку зображень з високою точністю і надійністю, і може використовуватися в різних додатках

комп’ютерного зору, таких як розпізнавання об’єктів, трекінг об’єктів, 3D моделювання і т.д. [7, 14].

1.3 Способи зіставлення структурних описів

Структурний опис, також відомий як множина дескрипторів, являє собою множину векторів із числових або бінарних значень, які описують об’єкт або його частину на зображенні. Аналіз структурного опису полягає в оцінці його здатності точно описувати об’єкти на зображенні, а також у визначенні релевантності шляхом зіставлення опису із набором еталонних описів.

Метрики використовуються для оцінки схожості між структурними описами і визначення їх релевантності. Вони вимірюють відстань або схожість між двома дескрипторами.

Статистичне уявлення – це один із найуніверсальніших інструментів у сучасному інтелектуальному аналізі даних, ключовою перевагою якого є можливість використання загальної інформації про властивості класів об’єктів, що розпізнаються, для більш ефективного обліку особливостей об’єктів у просторі ознак, що дозволяє отримати більш точну класифікацію об’єктів. Статистичне уявлення широко використовують у різних галузях, включаючи машинне навчання, комп’ютерне зір, штучний інтелект, обробку природної мови тощо. Воно стає все більш популярним завдяки своїй універсальності, надійності та точності [8, 10, 22].

Метрики відіграють важливу роль у вирішенні завдань розпізнавання образів та інтелектуального аналізу даних. Вони є кількісним показником, який дозволяє визначити приналежність елемента до множини або подібність множин один з одним. Значення метрики часто пов’язані з ймовірнісними характеристиками віднесення елемента до класу. Незважаючи на те, що на сьогоднішній день відомо багато метрик, завдання пошуку та розробки нових

метрик залишається актуальним. Основна мета створення нових метрик полягає у підвищенні ефективності вирішення нових практичних завдань [5].

Метрика Хеммінга (МХ) вимірює різницю між двома послідовностями однакової довжини, підраховуючи кількість позицій, де ці послідовності різняться. Якщо розглядати двійкові послідовності, то МХ обчислюється як кількість різних бітів у цих послідовностях. Наприклад, якщо є дві послідовності «1010101» і «1110000», то МХ між ними дорівнює 4, тому що вони різняться в чотирьох позиціях: на другому, третьому, п’ятому і сьомому бітах.

Однією з переваг МХ є те, є те, що МХ може легко узагальнена на випадок довільних алфавітів, де замість бітів можуть використовуватися символи з будь-якого алфавіту [1, 10].

Функція, якою задається МХ, у загальному вигляді задається так:

$$d_H(X_i, X_j) = sign \sum_{s=1}^{p} \left| x_i^{(s)} - x_j^{(s)} \right|. \quad (1.7)$$

Характеристики метрики притаманні відстані Хеммінга, коли вона відповідає наступним умовам:

а) $d_H(X_i,X_j) \geq 0$;

б) $d_H(X_i,X_i) = 0$;

в) $d_H(X_i,X_j) = d_H(X_j,X_i)$;

г) $d_H(X_i,X_k) \leq d_H(X_i,X_j) + d_H(X_j,X_k)$.

Міська метрика, або манхеттенська відстань (МВ), є способом обчислення відстані між двома точками в прямокутній системі координат. Воно отримало свою назву завдяки формулі, що нагадує маршрути переміщення у місті Манхеттен, де вулиці утворюють прямокутну сітку. Також ця метрика відома як «таксікаб-метрика» [12].

МВ між двома точками визначається як сума модулів різниць їх координат по осях *X* і *Y*. Іншими словами, якщо дві точки мають координати (x_1, y_1) і (x_2, y_2), то манхеттенська відстань між ними дорівнює:

$$| x1 – x2 | + | y1 - y2 |. \quad (1.8)$$

МВ відповідає сумі горизонтальної та вертикальної відстані між двома точками у прямокутній системі координат. У випадку, якщо точки знаходяться на одній горизонтальній лінії, відстань між ними дорівнює різниці їх координат по осі *X*. Якщо ж точки розташовані на одній вертикальній лінії, відстань між ними буде дорівнює різниці їх координат по осі *Y*.

МВ знаходить застосування у багатьох областях, включаючи геометричні обчислення, машинне навчання, КЗ тощо.

Можна виділити два різних типи метрик – метрики між дескрипторами та між множинами.

Метрики між дескрипторами оцінюють схожість або відстань між двома окремими дескрипторами. Наприклад, ви можете порівняти два дескриптори SIFT за допомогою евклідової відстані або косинусної схожості, щоб визначити, наскільки вони схожі.

Метрики між множинами використовуються для вимірювання відстані між двома наборами об’єктів, що знаходяться у різних множинах. Їх застосовують для порівняння, класифікації та кластеризації множин об’єктів. Коефіцієнт Жаккара та відстань Хеммінга є прикладами метрик між множинами [20, 21].

Отже, основна різниця полягає в тому, що метрики між дескрипторами порівнюють окремі об’єкти, тоді як метрики між множинами порівнюють цілі множини об’єктів. Крім того, метрики між множинами можуть залежати від конкретного способу обчислення подібності або відстані між множинами, так як це оцінюється на основі спільних та загальних елементів множини [8–11].

Приклад метрик між дескрипторами:

– Евклідова відстань: це метрика, яка обчислює відстань між двома точками в n-мірному просторі. Для двох векторів ознак A та B евклідова відстань можна записати як [8, 22]

$$dist = \sqrt{\sum_{i=1}^{n}(A_i - B_i)^2}\,, \tag{1.9}$$

де i = 1, 2, ..., n;

– косинусна відстань: це метрика, яка вимірює кут між двома векторами ознак. Косинусна відстань між векторами A та B обчислюється за формулою (1.10):

$$cos(\theta) = \frac{AB}{|A|\cdot|B|}, \tag{1.10}$$

де AB – скалярний добуток векторів;

$||A||$ та $||B||$ – довжини векторів.

Приклад метрики між множинами – коефіцієнт Жаккара. Ця метрика вимірює ступінь збігу між двома множинами. Вона обчислюється за формулою (1.11):

$$|A \cap B| \,/\, |A \cup B|, \tag{1.11}$$

де A і B – дві множини об’єктів.

Використання метрик, які враховують лише часткове представлення множини КТ об’єктів, дозволяє вирішувати ширший спектр завдань та підвищувати ефективність роботи систем розпізнавання. Це особливо важливо для забезпечення стабільності та надійності функціонування систем розпізнавання.

1.4 Постановка задачі

Створення моделей класифікації зображень, використовуючи методи аналізу структурних описів за допомогою критеріїв унікальності та інформативності є важливим та актуальним завданням.

Об’єтом роботи є методи класифікації зображень з відбором за критеріями унікальності та інформативності структурних ознак.

Метою роботи є створення класифікатора для вирішення проблеми швидкого віднесення зображень до класів та їх розпізнавання у системах комп’ютерного зору. За допомогою відбору частини дескрипорів за показниками інформативності та унікальності класифікатор забезпечує швидку та точну класифікацію зображень у реальному часі.

Для досягнення мети необхідно вирішити такі завдання:

– освоїти способи опису зображень та алгоритми їх обробки;

– розробити та реалізувати алгоритм класифікації зображень, який використовуватиме критерії унікальності та інформативності;

– написати програмний код для формування структури та зв’язків між ключовими точками на картинці із застосуванням методу BRISK;

– реалізувати алгоритм для класифікації вхідного зображення на основі розрахунку відстані до еталонів з фіксованої бази;

– протестувати роботу програмного коду, використовуючи тестовий матеріал у вигляді кольорових зображень NFT-персонажів розміром 420×420 пікселів;

– здійснити оцінку ефективності розробленого способу класифікації зображень.

2 КЛАСИФІКАЦІЯ ЗОБРАЖЕНЬ З ВИКОРИСТАННЯМ КРИТЕРІЇВ УНІКАЛЬНОСТІ ТА ІНФОРМАТИВНОСТІ

2.1 Огляд сучасних моделей класифікації

Хоча алгоритм класифікації зображень може відрізнятися залежно від методу, що застосовується, його етапи можна узагальнити в наступні кроки:

– для подальшої обробки зображень потрібна їхня попередня підготовка, що включає масштабування, нормалізацію, фільтрацію шуму, конвертацію в інший формат та інші подібні процедури;

– аналіз зображень з метою виділення ознак, необхідних для класифікації, включає виділення контурів, визначення колірних гістограм, вилучення текстурних ознак та інші подібні процедури;

– вибір класифікатора завдання класифікації зображень залежить від її специфіки, і найпопулярнішими методами класифікації є метод опорних векторів (SVM), НМ та дерева рішень;

– для навчання класифікатора з категоризації зображень потрібно розмічений набір даних, у якому алгоритм проходитиме аналіз ознак зображень виявлення закономірностей, здатних допомогти у розрізненні категорій;

– після закінчення навчання класифікатора його необхідно протестувати на новому наборі даних, не використаних раніше навчання, щоб оцінити його точність роботи під час тестування;

– успішно протестований класифікатор може бути застосований для класифікації нових зображень, де алгоритм аналізуватиме ознаки нового зображення та класифікуватиме його у відповідну категорію.

Є велика кількість сучасних мережних моделей класифікаторів зображень, деякі з яких є популярними і широко використовуються у сучасних програмах КЗ. Деякі з них – згорткові НМ (CNN), рекурентні НМ (RNN) і мережі довгої короткострокової пам’яті (LSTM), а також метод опорних

векторів (SVM) і дерева рішень [7–10, 22]. Окрім того, існують передбачувані моделі, які можуть бути налаштовані для вирішення специфічних задач класифікації зображень.

Загалом виділяють чотири існуючі категорії методів розпізнавання:

– порівняння із зразком (ПЗ);

– нейронні мережі (НМ);

– статистичні методи (СМ);

– структурні та синтаксичні методи (ССМ) [23].

У випадку з ПЗ в пам’яті машини зберігається множина образів, по одному з кожного класу, які будуть використовуватися для порівняння з невідомим вхідним класом. Класифікація відбувається з урахуванням заданого критерію подібності, де вхідний образ належить до класу, чий стандарт йому відповідає найкраще. Такий метод застосовується, наприклад, для розпізнавання друкованих літер та банківських чеків. Однак цей підхід має недолік – складність вибору еталона, який найкраще представляє кожен клас, та встановлення критерію відповідності. Ці складності виникають особливо при розпізнаванні рукописних літер, коли образи одного класу може бути значно спотворені. Найбільш просунутий спосіб полягає в використанні ознак, тобто деякої множини вимірювань, які малочутливі до змін та спотворень образів. За цих умов розпізнавання образів розглядається у двох задачах.

Перша задача полягає у виборі вимірювальних ознак для вхідного образу. Рішення про те, які виміри слід проводити часто є суб’єктивним і залежить від практичних обставин. На даний момент немає загальної теорії вибору вимірювальних ознак.

Друга задача розпізнавання образів полягає у класифікації вхідного образу з урахуванням відібраних ознак. Вона ґрунтується на ухваленні рішення про належність вхідного образу до певного класу [24].

НМ використовують для класифікації зображень шляхом імітації функції нейронів у головному мозку. Вони складаються з множини шарів, кожен з яких містить нейрони, що обробляють вхідні дані та передають вихідні

сигнали на наступний шар. Вхідні дані, представлені у вигляді пікселів зображення, проходять через усі шари, доки не досягнуто останній шар, який видає вихідні дані у вигляді ймовірності приналежності до кожного з можливих класів [5].

Навчання НМ відбувається шляхом налаштування ваги між нейронами в кожному шарі. На початку навчання ваги ініціалізуються випадковими значеннями, а потім коригуються за допомогою методу зворотного поширення помилки мінімізації помилки передбачення. Хоча НМ мають високу точність класифікації зображень, особливо для складних зображень з множиною ознак, для навчання моделі потрібен великий обсяг даних та обчислювальних ресурсів, і результати роботи моделі можуть бути складними для інтерпретації [6, 42].

Розглянемо процес розпізнавання об’єктів з використанням згорткової НМ (рис. 2.1). Вхідне зображення не має бути надто великим, бо тоді буде підвищуватися складність обчислення, а швидкість обробки зменшується. Але при цьому розмір і занадто маленьким не повинен бути, тому що в такому випадку не виявляться ключові точки.

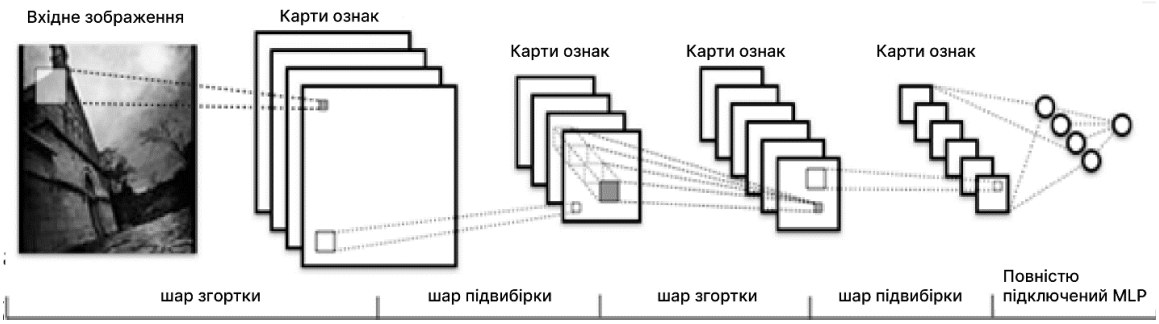

Рисунок 2.1 – Топологія згорткової НМ

Методом підбору вирішується завдання визначення розмірів зображення. Вхідний шар враховує двовимірну топологію зображень та складається з кількох матриць (карт). Якщо зображення представлено у відтінках сірого, то кількість матриць може дорівнювати одному, а у разі, якщо воно складається з трьох каналів (червоного, зеленого і синього), то матриць

може бути три. Кожна карта відповідає певному каналу зображення. Вхідні дані кожного значення пікселя нормалізуються в діапазоні від 0 до 1 відповідно до формули:

$$f(p, min, max) = \frac{p - min}{\max - min}\ , \qquad (2.1)$$

де f – функція нормалізації;

p – значення кольору пікселя від 0 до 255;

min – 0;

max – 255.

Шар згортки є сукупністю матриць ознак, де кожна карта має синаптичне ядро. Визначення оптимальної кількості матриць здійснюється експериментально. Зазвичай кожної матриці попереднього шару рекомендується використовувати дві матриці, у результаті виходить шість матриць в згортковому шарі. Розміри всіх матриць однакові і визначаються як (2.2), що забезпечує їхню узгодженість та знижує складність обчислень [43].

$$(w, h) = (mW - kW + 1, mH - kH + 1)\ , \qquad (2.2)$$

де (w,h) – обчислюваний розмір згорктової матриці;

mW – ширини матриці;

mH – висота матриці;

kW – ширина ядра;

kH – висота ядра.

Розмір ядра не повинен бути заанаадто великим чи маленьким, зазвий беруть у діапазоні від 3×3 пікселів до 7×7 пікселів.

Ядро – це головна особливість згорткової мережі, вона є єдиною системою вагових або синаптичних зв’язків. На відміну від звичайних багатошарових мереж, згорткові мережі мають загальні ваги, що дозволяє

скоротити кількість зв’язків і виявляти однакові ознаки по всій області зображення. На кожному згортковому шарі початкові значення карти дорівнюють 0, а значення ядра ваги задаються випадковим чином в діапазоні від -0,5 до 0,5. Ядро ковзає попередньою картою, використовуючи операцію згортки для обробки зображень за формулою:

$$(f \times g)[m,n] = \sum_{k,l} f[m-k, n-l] \times g[k,l], \quad (2.3)$$

де f – початкова матриця зображення;

g – ядро згортки.

Результат обробки початкової матриці може мати розмір, який менше, дорівнює або більше розміру вихідного зображення, залежно від методу обробки країв, як показано нижче (рис. 2.2).

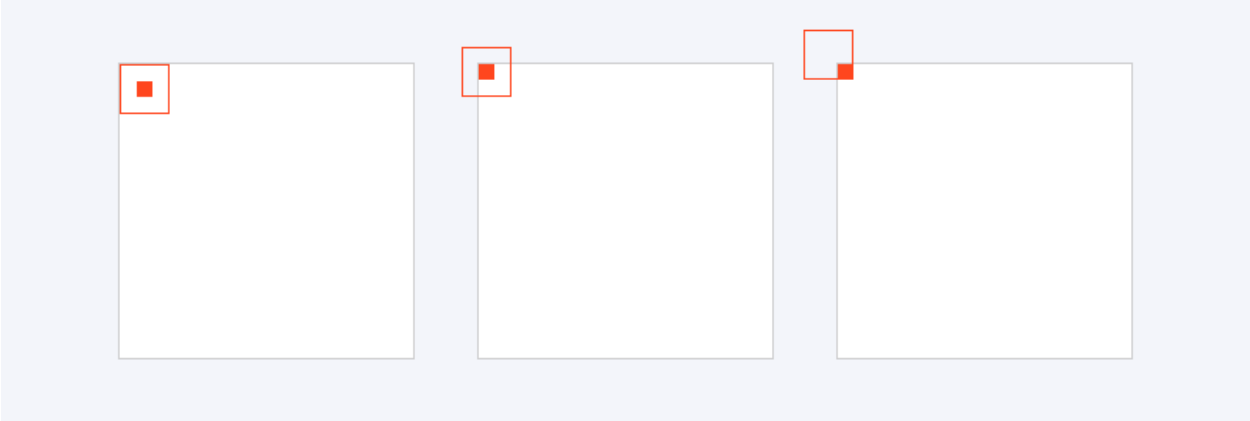

Рисунок 2.2 – Типи згортки початкової матриці

Підвибірковий шар скорочує розмірність карт попереднього (згорткового) шару, використовуючи матриці, їх кількість відповідає попередньому шару, і становить 6. Фільтрація непотрібних деталей допомагає запобігти перенавченню мережі. Ядро розміром 2×2 сканує карту попереднього шару, зменшуючи її в два рази. Відмінність від згорткового шару в тому, що ядро, що сканує, не перетинається [44].

Повнозв’язковий шар - це останній тип шару нейронної мережі, який працює як звичайний багатошаровий перцептрон. Його основне завдання – класифікація. Модель складної нелінійної функції оптимізується, щоб покращити якість розпізнавання.

Статистичні методи розпізнавання образів базуються на використанні статистичних засобів для аналізу даних та виведення узагальнених закономірностей. Ці методи дозволяють зібрати узагальнену інформацію про властивості різних класів об’єктів та використовувати її для класифікації нових об’єктів [5].

Один з основних статистичних методів розпізнавання образів – байєсівський класифікатор. Він ґрунтується на формулі Байеса, яка використовується для обчислення умовної ймовірності належності об’єкта до певного класу при заданому наборі ознак. Формула Байеса має наступний вигляд:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}, \tag{2.4}$$

де *P(A|B)* – умовна ймовірність того, що об’єкт належить до класу *A*, заданого набору ознак *B*;

P(B|A) – ймовірність спостереження набору ознак *B* за умови, що об’єкт належить до класу *A*;

P(A) – апріорна ймовірність того, що об’єкт належить до класу *A*;

P(B) - ймовірність спостереження набору ознак *X*.

Використовуючи (2.4), можна порівнювати умовні ймовірності належності об’єкта до різних класів та класифікувати його до того класу, для якого ймовірність найвища.

Метод наївного Байєса – це один із найпоширеніших методів машинного навчання для завдань класифікації, заснований на імовірнісній моделі. Він може бути використаний для класифікації зображень, де кожне зображення

розглядається як набір випадкових змінних, які описують можливість приналежності зображення до певного класу.

Для класифікації зображень метод наївного Байєса використовує дані про розподіл пікселів у кожному зображенні, щоб визначити ймовірність приналежності зображення до кожного класу (рис. 2.3). На відміну від методу дерев рішень, у методі наївного Байєса розглядається кожен піксель окремо, а не всі пікселі разом, що підвищує точність класифікації.

Метод наївного Байєса швидко навчається на великих обсягах даних та може швидко класифікувати нові зображення. Крім того, він працює добре з різними типами ознак, включаючи категоріальні та числові [20].

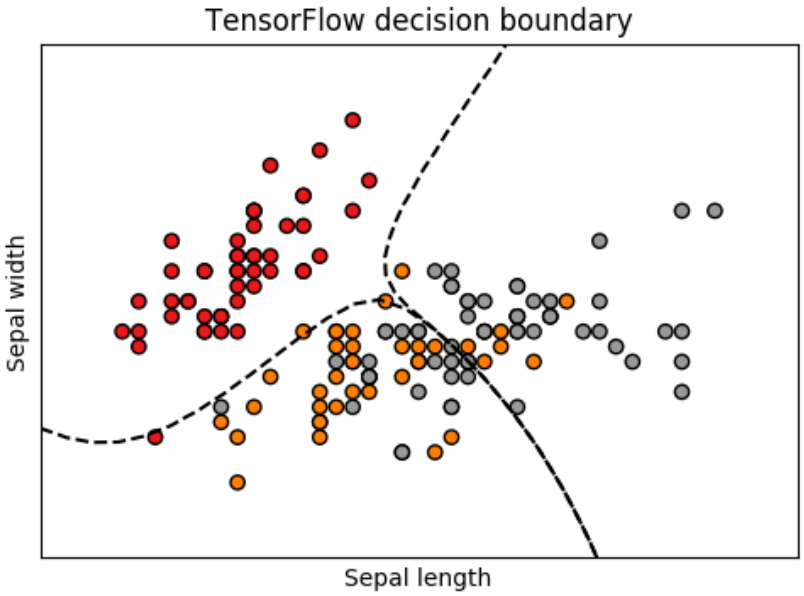

Рисунок 2.3 – Класифікатор наївного Байєса у TensorFlow

Однак метод наївного Байєса має й недоліки. Він передбачає, що кожна ознака у зображенні незалежна від інших ознак. Також метод наївного Байєса не враховує контекст зображення, як-от положення об’єктів на зображенні. В результаті він може мати обмежену точність класифікації в порівнянні з більш складними методами, такими як нейронні мережі.

Структурні методи в теорії розпізнавання образів використовуються для визначення структури об’єктів на основі їх властивостей, що дозволяє

покращити якість та ефективність розпізнавання. Ці методи базуються на аналізі геометричних та топологічних характеристик об’єктів і намагаються встановити структуру об’єкта шляхом порівняння його з попередньо заданою структурою.

Один з підходів до структурного розпізнавання – це синтаксичне розпізнавання (syntactic pattern recognition). Синтаксичне розпізнавання ґрунтується на теорії формальних мов та автоматів, яка описує мови та граматики, що використовуються для опису структури об’єктів.

Одним з підходів до структурного розпізнавання образів є метод структурного зіставлення (structural matching). Цей метод базується на порівнянні структури об’єкта з попередньо заданою моделлю шляхом зіставлення складових частин та їх відношень [32].

Для структурного розпізнавання образів можна використовувати алгоритми, які базуються на вимірюванні відстані між об’єктами в просторі характеристик. Ці характеристики можуть бути такі, як розмір, форма, кольорова гама, текстура тощо. Зазвичай використовують методи

Одним з прикладів алгоритму структурного розпізнавання є алгоритм дерева пошуку (tree search algorithm). Цей алгоритм використовується для розпізнавання образів, які можуть бути представлені у вигляді дерева з вузлами та гілками. Алгоритм порівнює дерево образу з деревом моделі та знаходить оптимальний шлях з відповідностями між вузлами.

Метод опорних векторів (МОВ) можна використовувати для класифікації зображень як метод машинного навчання. Він заснований на пошуку гіперплощини, яка розділяє зображення різних класів найефективнішим способом. Для цього МОВ шукає межу ухвалення рішення між двома класами даних, яка максимізує відстань між найближчими точками різних класів – опорними векторами (рис. 2.4) [22, 23].

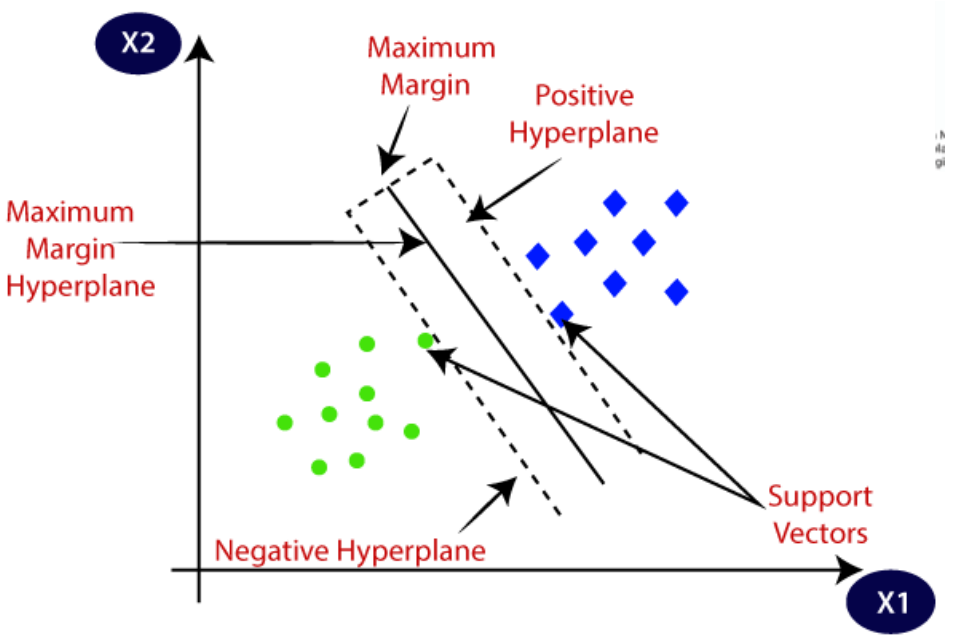

Рисунок 2.4 – Ілюстрація класифікації ознак за МОВ

Для класифікації зображень метод МОВ починає з отримання ознак зображення, які потім використовуються для створення вектора ознак. МОВ визначає якого класу відноситься кожен вектор ознак в навчальному наборі даних і будує межу прийняття рішення, що розділяє класи на площині.

Після навчання МОВ може бути використаний для класифікації нових зображень. Під час класифікації МОВ використовує межу прийняття рішення, яку він побудував під час навчання, щоб визначити якого класу відноситься вектор ознак нового зображення.

МОВ може працювати з великою кількістю ознак і вчитися на невеликих навчальних наборах даних, але для навчання на великих наборах даних можуть знадобитися значні обчислювальні ресурси та час.

Метод дерев рішень – один із найпоширеніших у машинному навчанні для класифікації та регресії. Він полягає у побудові дерева, де кожен вузол – це рішення про вибір ознаки, на основі якої дані будуть поділені на класи. У класифікації зображень дерево може визначати класи кожного зображення, використовуючи вектор ознак, наприклад, яскравість пікселів.

Після побудови дерева нові зображення класифікуються, починаючи з кореня, і кожному вузлі приймається рішення з урахуванням ознак [28].

K-найближчих сусідів (KNN) – це метод класифікації зображень, який використовує найближчих сусідів визначення класу нового зображення. Він обчислює відстань між об’єктами та вибирає *k* найближчих зображень з навчальної вибірки. Потім він використовує мітки класів найближчих сусідів для ухвалення рішення про класифікацію нового зображення (рис. 2.5). Метод KNN може працювати з наборами даних будь-якого розміру та обробляти як категоріальні, так і числові ознаки, що робить його простим та зрозумілим у використанні. Однак, при обробці великих навчальних наборів даних пошук найближчих сусідів може зайняти багато часу, а метод також схильний до помилок при роботі з шумними даними. Необхідно також правильно вибирати значення параметра *k* для досягнення найкращої точності класифікації [5].

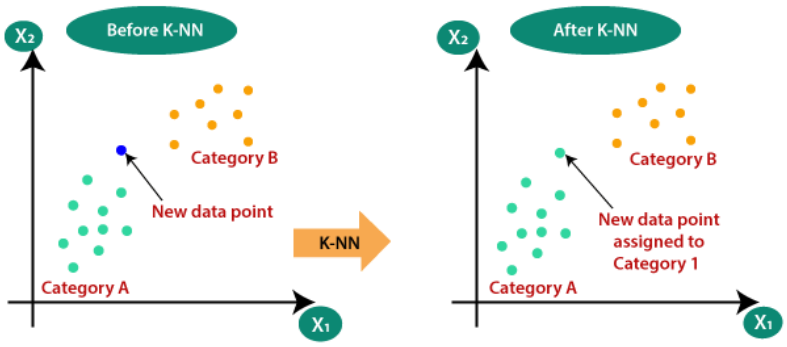

Рисунок 2.5 – Схематичне зображення роботи алгоритму KNN

Для пояснення роботи алгоритму K-NN використовують такі кроки:

а) виберіть кількість сусідів *K*;

б) обчисліть евклідову відстань для *K* найближчих сусідів;

в) виберіть *K* найближчих сусідів на основі обчисленої евклідової відстані;

г) підрахуйте кількість точок даних у кожній категорії серед цих *K* сусідів;

ґ) віднесіть нові точки даних до категорії, яка має найбільшу кількість сусідів;

д) модель готова до використання [25].

Метод випадкового лісу (Random Forest) є одним із найпопулярніших алгоритмів класифікації зображень. Цей метод ґрунтується на створенні множини дерев рішень та об’єднанні їх голосуванням. Для створення випадкового лісу спочатку будується набір дерев рішень. Кожне дерево будується на основі підмножини навчальних даних, які вибираються випадковим чином. Це дозволяє зменшити кореляцію між деревами та збільшити різноманітність ансамблю.

Кожне дерево рішень розбиває простір на регіони за допомогою послідовного застосування умов на значеннях ознак. Для кожного регіону визначається клас, що найчастіше зустрічається серед навчальних даних, що потрапляють у цей регіон. Це називається листом дерева. Коли необхідно класифікувати нове зображення, воно проходить через кожне дерево рішень і кожне дерево видає свою відповідь. Потім відбувається голосування і підсумкова відповідь вибирається на основі відповідей усіх дерев.

Однією з головних переваг методу випадкового лісу є його здатність обробляти великі обсяги даних та нечислові ознаки. Крім того, він має високий ступінь стійкості до шуму та перенавчання.

Глибоке навчання використовує глибокі нейронні мережі для отримання ознак із зображень та класифікації [43-45]. На відміну від інших методів, де ознаки витягуються вручну, глибоке навчання дозволяє мережам самостійно отримувати ознаки. Цей метод використовує множину верств, де кожен шар витягує певні ознаки. Зворотне поширення помилки використовується для оптимізації ваги мережі. Глибоке навчання має високу точність, може використовуватися для зображень різних розмірів та форматів, але потребує великих обчислювальних ресурсів та може бути схильним до перенавчання.

2.2 Застосування параметрів унікальності та інформативності

BRISK може містити сотні векторів, що суттєво збільшує час обробки. Задача полягає в скороченні кількості векторів за допомогою стисненого опису O^* з урахуванням відображення: $\Omega: O \to O^*$.

У теорії розпізнавання зменшення множини ознак є способом стиснення даних з метою зниження витрат на розпізнавання. Цей процес називається формуванням підмножини «значущих», «унікальних» або «інформативних» ознак [1, 34].

Критерієм, за яким оцінюють подібність елементів B_1^n є метрика ρ, за яку найчастіше беруть Евклідову метрику з B_1^n. За критерій еквівалентності КТ o_i, o_j беруть значення $\rho(o_i, o_j)$, яке не перевищує апріорну величину порогу δ_o [26, 41].

Дві ключові точки еквівалентні, якщо умова $\rho(o_i, o_j) \leq \delta_o$. Подібність значень характерних ознак усередині опису та між описами оцінюється евклідовою відстанню між векторами o_i, o_j:

$$\rho(o_i, o_j) = \sqrt{\sum_v [o_i(v) - o_j(v)]^2} \quad . \tag{2.5}$$

Або як різниця числа бітів, які відрізняються між двійковими представленнями, якщо за метрику взята Хемінгова відстань (рис. 2.6).

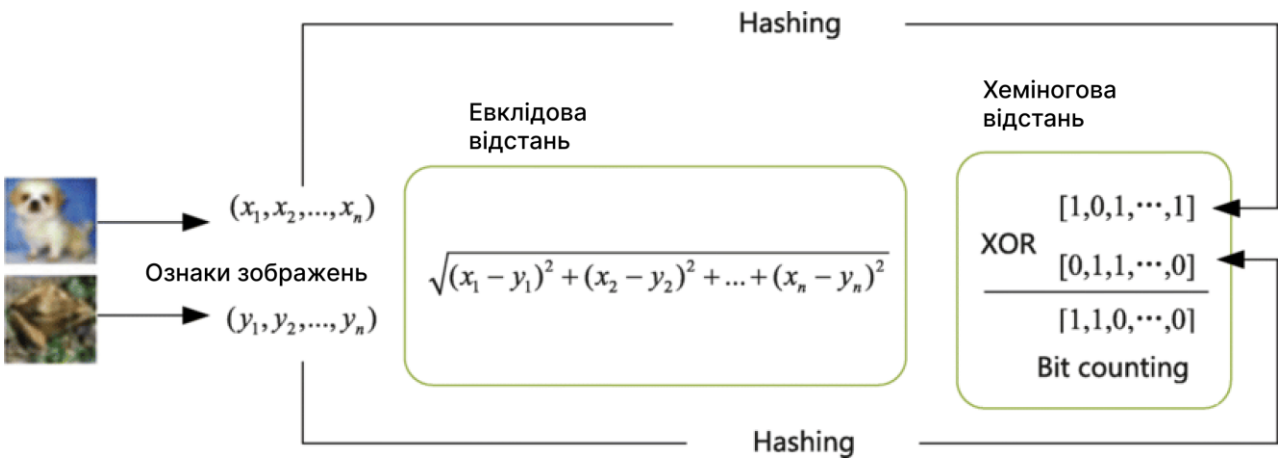

Рисунок 2.6 – Різниця між Евклідовою та Хемінговою відстанями

Оцінити ступінь подібності між ознаками можна, побудувавши відношення на множині їх характеристик у межах опису та бази описів. Використання цього методу дозволяє покращити розпізнавання за рахунок урахування унікальних властивостей ознак. Порогове значення δ_o окреслюється відсоток максимального можливого значення (точності).

Вибір δ_o залежить також від процедури обробки. Величини порогу δ_o можуть відрізнятись. Наприклад, у процедурах навчання, під час встановлення еквівалентності ключових точок, при класифікації. Встановлене відношення еквівалентності елементів O при фіксованому значенні δ_o визначає розбиття O.

Гранулювання інформації є необхідною властивістю інтелектуальних систем, оскільки вони використовують семантику для формалізації вирішення завдань. Чітке та нечітке гранулювання є важливими інструментами для обробки даних. Гранулою інформації є підмножина універсуму, на якому визначено відношення подібності або еквівалентності між елементами. Гранулою може бути об'єднанням будь-якої кількості атомарних елементів, а універсум або опис можна представити як множину гранул. Основи теорії міри та відношень розроблені з використанням гранул, що дозволяє вирішувати завдання зі зниженням розмірності даних та ускладненням взаємозв'язків між ними. Ступінь гранулювання може бути обчислений як сума (інтеграл) значень функції приналежності елементів, що робить гранулювання корисним інструментом для обробки та аналізу даних [1, 41].

Міра $d(A)$ дискретної гранули A визначена так:

$$d(A) = \sum_{a \in A} \mu_A(a), \tag{2.6}$$

де $\mu_A(a)$ – значення функції належності гранулі A, $\mu_A(a) \in [0,\ 1]$.

Для чіткого подання $\mu_A(a)$ набуває бінарних значень 0 або 1. Операції над гранулами відповідають теорії множин, а гранули можуть формувати ієрархічні структури та утримувати один одного. Гранулювання даних

залежить від параметрів та реалізується певним методом. Також можна розглядати питання про оптимальне гранульування на основі нечіткої логіки.

Для кожного еталону $O(q)$ з множини Q застосовується функція належності $\mu_A(a)$. Для кожної КТ підраховується відстань між її векторним описом (дескриптором) та іншими векторами характерних ознак. ХО вважаються однаковими, якщо відстань між їх векторами дескрипторів менша за певне значення допустимої помилки, що залежить від бази еталонів [1, 26].

Функція належності передбачає залежність всіх еталонів одного класу один від одного. Одна характеристика може бути унікальною у межах одного класу, але повторюватися у межах інших класів. Додавання нового еталону до класу вимагає перерахунку належності дескрипторів всіх еталонів у межах цього класу. Аналіз за допомогою функції належності здійснюється повторно.

Коефіцієнт унікальності τ кожної ХО обчислюється наступним чином:

$$\tau = e/total, \tag{2.7}$$

де e – кількість однакових унікальних ознак у межах одного еталону $O(q)$;

$total$ – загальна кількість еквівалентних унікальних ознак бази еталонів Q.

Якщо значення дескриптора менше певного порогу δ_o, він не враховується. Якщо ця ключова точка зустрічається в інших еталонах з бази еталонів Q, вона автоматично виключається. Видалення таких «неунікальних» ключових точок з множини досліджуваних зображень може зменшити обсяг бази даних і підвищити швидкість розпізнавання зображень.

Якість та швидкодія розпізнавання покращується при $\tau \to 1$, а обсяг бази даних зменшується.

Для елементу o_i опису $O = \{o_i\}_{i=1}^{I}$ визначимо число h_i повторень:

$$h_i = \sum_{j=\overline{1,I}} 1(\rho(o_i, o_j) \le \delta_o), \tag{2.8}$$

де $1(\rho(o_i, o_j) \le \delta_o) = \begin{cases} 1, & \rho(o_i, o_j) \le \delta_o, \\ 0, & \text{otherwise.} \end{cases}$;

$h_i \in C_+$ – це число елементів O, еквівалентних елементу $o_i \in O$ з точністю до δ_o відповідно істинності предикату $1(\rho(o_i, o_j) \le \delta_o)$, C_+ – множина цілих невід'ємних чисел.

Розглянемо скінченну множину $O = \{O^i\}_{i=1}^{Q}$ описів, що відповідають базі зображень з Q еталонів. Для кожної ключової точки $o_{ij} \in O^i$ еталону з номером i обчислимо індекси унікальності: α_{ij}, β_{ij}:

$$\alpha_{ij} = c_{ij}/s_i, \tag{2.9}$$

$$\beta_{ij} = c_{ij}^{s}/(s - s_i), \tag{2.10}$$

де $\alpha_{ij} \in [0,\ 1]$, $\beta_{ij} \in [0,\ 1]$, c_{ij} – число подібних елементів для o_{ij} в i-му еталоні;

s_i – число характерних ознак i-го еталону;

c_{ij}^{s} – значення для o_{ij} в базі еталонів, крім еталону з номером i [1] ;

$s = \sum_i s_i$ – загальна кількість елементів описів бази O.

Для елементів кожного еталону обчислимо значення індексів (2.10) та інтегральні показники $\sum_j \alpha_{ij}, \sum_j \beta_{ij}$. У випадку $s_i \ne s_q$ при $i \ne q$.

Індекс α_{ij} показує ступінь повторюваності j-го елементу всередині i-го еталону. Індекс β_{ij} показує рівень повторюваності j-го елементу в описах інших об'єктів бази, крім i-го. Найбільш значущим для розпізнавання можна вважати β_{ij}. Чим ближче β_{ij} до нуля, тим більш унікальний елемент серед інших елементів бази. Сума $\alpha_{ij} + \beta_{ij}$ показує ступінь унікальності елемента по всій базі. Інтегральні показники $\sum_j \alpha_{ij}$, $\sum_j \beta_{ij}$ характеризують загальну унікальність елементів еталону в межах бази.

Проведення аналізу значень $\{o_{ij}, \alpha_{ij}, \beta_{ij}\}$ та інтегральних показників дозволяє оцінити ступінь відмінності об'єктів бази під час використання

структурної класифікації на практиці. Вивчення індексів унікальності α_{ij}, β_{ij} дозволяє розробити та використовувати процедури скорочення характеристик як усередині еталона, так і в рамках усієї бази даних [26].

Процес фільтрації здійснюється шляхом створення предикату, який визначає важливість кожного елемента в новому описі. Використовуючи отримані характеристики, можна виділити групи ключових точок, які рідко чи часто повторюються, та створити гранули.

Можна виділити наступні способи редукції ознак:

– кожен еталон обробляється окремо, щоб знайти кортеж з ознак g з найменшими значеннями α_{ij};

– для кожного еталона формується кортеж з ознак g, унікальних серед інших еталонів, тобто з найменшими значеннями β_{ij}.

Збільшення швидкодії розпізнавання зображень для цих способів становить s_i/g. Трансформація ознакового простору здійснюється на попередньому етапі та не впливає на час розпізнавання.

Другий спосіб перспективніший і впливає на ефективність. Критерій β_{ij} показує рівень відмінності еталонів. Ознаки з високим значенням α_{ij} інтенсивно повторюються всередині опису. Ключові точки з максимальним значенням α_{ij} містять інформаційну складову та відображаються на результатах розпізнавання [1].

За допомогою визначення гранул і методів обчислення індексів унікальності можна створювати елементи опису в системі класифікації, що дозволяє групувати елементи гранули і значно прискорює процес їх обробки.

Класи можуть бути визначені на основі характеристик α_{ij}, β_{ij}.

Класифікуючи різні елементи в базі $U = \{u_1, u_2, \dots, u_i, \dots, u_l\}$, отримаємо подання еталону в вигляді мультимножини. Належність точки o мультимножині A встановлюється шляхом обчислення характеристичної функції:

$$\chi_A(Z) = \begin{cases} 1, & o \in A, \\ 0, & o \notin A. \end{cases} \quad (2.11)$$

Перевірка $o \in A$ виконується порівнянням o з елементами U на основі $\rho(o, u_i)$ (або іншої міри).

Клас u_* елементу o визначимо шляхом дискретної оптимізації:

$$u_* = arg \min_{u_i \in U} \rho(o, u_i), \ \rho(o, u_*) \leq \delta_o, \quad (2.12)$$

на множині U . Класифікація (2.12) включає перевірку умови $\rho(o, u_*) \leq \delta_o$.

Запропонований підхід дозволяє видалити з розгляду окремі елементи зображення.

На основі (2.12) при k базових класах для еталону $O(q)$ отримаємо векторне подання:

$$\phi(j) = \{\phi_1, \phi_2, \dots, \phi_i, \dots, \phi_k\}, \phi_i \in C_+, \quad (2.13)$$

де ϕ_i – кратність входження i -го базового елементу.

На основі запропонованої класифікації (2.12) та векторного опису (2.13) виконана трансформація $O \to \phi$. Дане перетворення здійснено з множинного подання у векторному описі ϕ кінцевої розмірності k з компонентами із C_+.

«Представники» кластерів задаються апріорно множиною U. Спосіб класифікації на основі U аналогічний розкладанню по системі ортогональних функцій. Основна відмінність полягає в тому, що для елементів U ортогональність не потрібна. В отриманій системі ознак не потрібно відновлювати елементи.

Розпізнавання об’єкта при мультимножинному поданні здійснюється шляхом обчислення та мінімізації відстані:

$$\rho(A, B) = \frac{\sum_i w_i |\phi_A(u_i) - \phi_B(u_i)|}{\sum_i w_i \, max[\phi_A(u_i), \phi_B(u_i)]}, \quad (2.14)$$

де w_i – вагові коефіцієнти (у даному дослідженні прийнято $\sum_i w_i = 1$);

$\phi_A(u_i), \phi_B(u_i)$ – кратності при поданні множин A та B у проєкції базового елементу $u_i \in U$ [1].

2.3 Застосування голосування у структурних методах

Метод голосування дескрипторів – це метод класифікації зображень, що реалізує принцип «дескриптор об'єкта – еталон». Він ґрунтується на порівнянні дескрипторів різних об'єктів, які розглядаються як набори векторних значень, що характеризують особливості об'єкта на зображенні. Спочатку систему завантажуються зразки – набори зображень, які стосуються певним класам. До кожного зображення обчислюються дескриптори [1, 5, 41-46].

Потім при класифікації нового зображення обчислюються його дескриптори. Для кожного дескриптора знаходиться найбільш схожий дескриптор у кожному зразку, з використанням метрики подібності (використовуємо Хеммінгову відстань).

Для кожного дескриптора нового зображення визначається найближчий дескриптор еталона з використанням різних метрик відстані, таких як відстань евкліда або косинусне відстань. Потім для кожного зразка підраховується кількість дескрипторів, які були класифіковані як найближчі до його дескрипторів. Нарешті, з усіх стандартів вибирається той, який отримав найбільшу кількість голосів, тобто найбільшу кількість найближчих дескрипторів. Саме цей еталон вважається класом, до якого належить нове зображення.

Метод голосування дескрипторів є простим і швидким способом класифікації, який може використовуватися в багатьох областях, наприклад, комп'ютерний зір і розпізнавання образів. Однак він не завжди є точним і

може давати помилкові результати, особливо якщо зразки погано представляють об’єкти певного класу.

До кожного еталону у методі голосування формується незалежна система кластерів, що дозволяє описати його ключові особливості. Кожен кластер у цій системі описується деякою ознакою, наприклад, відстанню між ключовими точками, кутом повороту об’єкта або кольором. Для кожного еталону визначається, які кластери найкраще описують нове зображення. Потім проводиться голосування, і вибирається еталон, у якого найбільше збігаються кластерів з дескриптором нового зображення (рис. 2.7).

Для визначення класу об’єкта за числом голосів компонентів, спочатку обчислимо максимальне значення для кожного рядка матриці:

$$c_v = arg \max_{i=1,\dots,N} \{d_v(i)\} \tag{2.15}$$

тобто визначимо окремо для кожного дескриптора опису найбільш вагомого класу за вектором розподілів, що відповідає параметру моди.

Для кожного дескриптора z_v опису Z за виразом (2.15) за значенням c_v інкрементуємо число h_i голосів елементів, віднесених до i-го класу:

$$h_i = \begin{cases} h_i + 1, & c_v = i, \\ h_i, & c_v \neq i. \end{cases} \tag{2.16}$$

За результатом визначення (2.16) для всієї множини дескрипторів опису Z отримуємо вектор голосів:

$$h = (h_1, h_2, \dots, h_N), \tag{2.17}$$

на підставі якого визначимо номер класу [1]:

$$j = arg \max_b h_b. \tag{2.18}$$

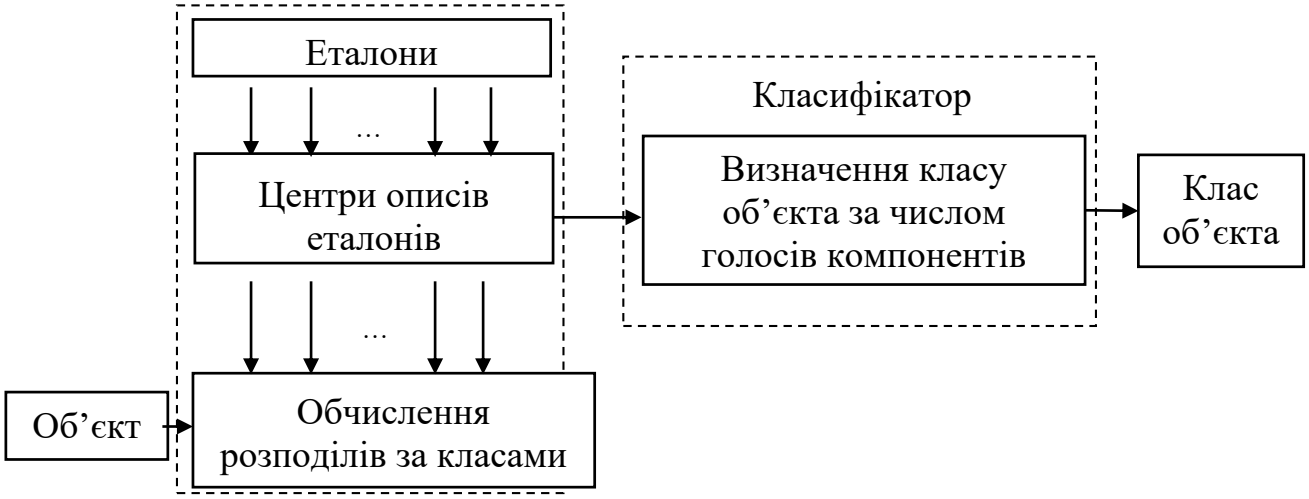

Рисунок 2.7 – Схема класифікації за методом голосування

Метод голосування є ефективним підходом до класифікації зображень, оскільки він дозволяє використовувати інформацію про безліч еталонів для ухвалення рішення про класифікацію нового об’єкта. Він також може бути розширений для роботи з великою кількістю еталонів та дескрипторів, що дозволяє підвищити точність класифікації.

3 ПРОГРАМНЕ МОДЕЛЮВАННЯ КЛАСИФІКАТОРА ЗОБРАЖЕНЬ

3.1 Обґрунтування вибору середовища програмної реалізації

У рамках кваліфікаційної роботи було змодельовано алгоритм для класифікації зображень на базі критеріїв унікальності та інформативності для структурних ознак.

C# – мова програмування, створена для переваги та вдосконалення мови C. Він не просто оновлює C, а піднімає його на новий рівень. Його назва, позначена символом #, асоціюється з музичним дієзом, що підвищує ноту на півтону. Таким чином, C# можна розглядати як покращену версію мови C.

У 1995 році компанія Sun Microsystems випустила мову програмування Java, яка була заснована на граматиці C++. Однак Microsoft, прагнучи перевершити Sun, вирішила не просто використовувати Java, а створити щось потужніше. Так виник Microsoft.NET Framework.

.NET не тільки впровадив проміжний байт-код, як у Java, а й розробив власну мову програмування – C#. C# став мовою, здатною перевершити свого попередника, надаючи розробникам більш гнучкі та потужні інструменти [46].

C# став мовою-флагманом платформи .NET і набув величезної популярності у світі розробки програмного забезпечення. Він забезпечує високу продуктивність, безпеку та масштабованість, а також широкий набір інструментів та бібліотек для розробки різноманітних програм. C# став основним вибором розробників, які прагнуть створювати сучасні та інноваційні рішення.

C# та C++ — дві мови програмування, які мають загальний корінь у мові C, але відрізняються за низкою аспектів. C# є еволюцією C, додаючи до неї об’єктно-орієнтовану парадигму з класами та об’єктами. Він забезпечує зручний синтаксис для створення класів з їх властивостями та реалізує основні принципи ОВП, такі як спадкування, інкапсуляція та поліморфізм.

Однією з ключових відмінностей C# від C++ є строга типізація, а також вбудована система збирання сміття, яка автоматично звільняє пам’ять та усуває необхідність вручну керувати пам’яттю. C# були додані такі здібності, як динамічне зв’язування, асинхронні способи і шаблони.

Однак основна відмінність між C + + і C # полягає в рівні складності та цілях. C++ є складнішим мовою, який пропонує покажчики на змінні і вимагає глибшого розуміння внутрішнього устрою мови. На відміну від C++, C# компілюється у проміжний байт-код, а не безпосередньо в машинний код. C# також тісно пов’язана з платформою .NET, у той час як C++ використовується в різних ситуаціях, подібно до універсального інструменту.

Ще однією відмінністю є відсутність C# множинного успадкування класів, що означає, що клас може бути нащадком відразу кількох різних класів. Це рішення було прийнято у C# з метою забезпечити простоту мови.

У цілому синтаксис C# вважається зручнішим проти C++. C# пропонує набір готових модулів на вирішення завдань, тоді як написання коду C++ вимагає глибшого розуміння мови.

C# і Java є схожими та головними конкурентами один одному. Вони розвивають концепції мови C++, спрощуючи її використання. Філософія цих двох мов дуже близька, але є технічні відмінності, які їх відрізняють одна від одної. Ці відмінності пов’язані з тим, що C# був розроблений спеціально для екосистеми Microsoft, тоді як Java замислювалася як відкрита платформа з відкритим кодом. Наприклад, різні підходи до реалізації параметризованих типів і відсутність C# перевірених винятків.

C# має низку переваг для вирішення завдань КЗ. Він пропонує велику бібліотеку OpenCV, яка надає широкий спектр функцій та алгоритмів для обробки зображень, розпізнавання об’єктів, трекінгу руху та інших завдань КЗ. Використання OpenCV разом із C# полегшує розробку складних алгоритмів КЗ [29].

C# інтегрується з .NET Framework, основною платформою, що забезпечує потужну інтеграцію коїться з іншими компонентами .NET. Це

дозволяє розробникам легко використовувати функціональні можливості .NET для обробки зображень, взаємодії з базами даних, створення інтерфейсу користувача та інших завдань, пов’язаних з КЗ.

C# має зручний синтаксис, що спрощує розробку та підтримку коду для завдань КЗ. Завдяки популярності C# у розробницькій спільноті існує активна спільнота, яка надає ресурси, форуми та бібліотеки для підтримки розробки завдань КЗ.

C# також надає механізми для роботи з багатопоточністю та асинхронним програмуванням. Це особливо корисно при обробці великих обсягів даних у реальному часі, що часто потрібне у завданнях КЗ. Розробники можуть використовувати паралельні обчислення та асинхронні операції для підвищення продуктивності та чуйності своїх додатків.

Крім того, існують сторонні бібліотеки та фреймворки, такі як Emgu CV, Accord.NET та AForge.NET, які розширюють можливості та надають додаткові алгоритми для роботи з КЗ у C#.

Важливо, що C# також має можливості інтеграції з іншими технологіями, включаючи машинне навчання та штучний інтелект. Розробники можуть використовувати бібліотеки машинного навчання, такі як TensorFlow.NET або ML.NET, для створення моделей КЗ та їх інтеграції до програм C#. Це дозволяє застосовувати сучасні методи та алгоритми машинного навчання для вирішення складних завдань КЗ.

В цілому, C# пропонує розробникам потужну та гнучку мову програмування для роботи із завданнями КЗ. Він має зручний синтаксис, широкі можливості інтеграції з іншими технологіями, підтримкою розробників та інструментами розробки, що робить його привабливим вибором для розробки додатків, пов’язаних з КЗ.

Резюмуючи, хотілося б навести приклад кілька всесвітньо відомих продуктів і сервісів, створених на C#. Більшість програм та сервісів, пов’язаних із продуктами Microsoft, розроблені саме на цій мові програамування. Наприклад, відомий сайт stackoverflow.com використовує C#

та фреймворк .NET для своєї розробки. Також створюються сайти Microsoft і Dell.

C# широко застосовується розробки десктопних додатків під операційну систему Windows. Програми, такі як Microsoft Visual Studio та Paint.NET, написані на C#. Бібліотека Windows Forms дозволяє створювати програми, такі як Skype, Microsoft Office та Photoshop.

Геймдев-розробники активно використовують двигун Unity для створення комп'ютерних ігор як у 2D, так і у 3D форматах. Програмування в Unity переважно здійснюється на C#. Деякі відомі ігри, такі як Rust, Hearthstone та Fall Guys створені в Unity з використанням C#. Крім того, C# дозволяє взаємодіяти з графічним та звуковим компонентами DirectX, що широко застосовуються в іграх [18].

C# також використовується для створення мобільних додатків. Існують платформи, такі як Xamarin, які дозволяють запускати код C# на різних операційних системах, включаючи Android і iOS. Багато популярних додатків написані на Xamarin.

У 2018 році до програмного середовища .NET було додано бібліотеку ML.NET, яка забезпечує можливості використання моделей машинного навчання. Хоча документації на ML.NET може бути трохи менше порівняно з бібліотеками на мові Python, вона зручна та успішно застосовується у виробничому середовищі.

Для реалізації було обране середовище Visual Studio Code (VSC) .

VS Code – одне з найпоширеніших інтегрованих середовищ розробки (ICP) у світі. Редактор підтримує підсвічування синтаксису для великої кількості мов програмування, автоматичне додавання відступів, автодоповнення коду, можливість швидкого переходу до визначення функцій та змінних. Крім того, всередині VS Code є інтегрована система контролю версій, яка дозволяє керувати та відстежувати зміни коду [7, 24].

У VS Code ви знайдете велику колекцію розширень, які дозволять вам налаштувати та розширити функціональність редактора відповідно до ваших

потреб. Ви можете додавати розширення для підтримки різних мов програмування, інтеграції з інструментами розробки, зміни в оформленні та інші додаткові можливості.

У редакторі є інтеграція із системою контролю версій Git, що значно полегшує роботу з репозиторіями та виконання різних команд Git. Ви зможете виконувати операції комміту, розгалуження, злиття та переглядати історію змін прямо з редактора, що робить процес керування версіями зручнішим та ефективнішим.

VS Code забезпечує інтуїтивний інтерфейс для налагодження коду. В рамках цього інтерфейсу ви можете легко налаштовувати точки зупинки, відстежувати значення змінних, виконувати кроки налагодження та використовувати безліч інших функцій, які допоможуть вам ефективно досліджувати та виправляти помилки у вашому коді.

VS Code має інтеграцію з різними засобами складання та розгортання, такими як npm, Docker, Azure та іншими. Завдяки цій інтеграції процес розробки, тестування та розгортання вашої програми стає більш зручним та спрощеним. Ви зможете легко використовувати ці інструменти прямо всередині VS Code, що дозволить вам ефективно керувати та налаштовувати середовище розробки та швидко виконувати необхідні операції складання та розгортання вашої програми.

Ще однією перевагою є можливість скористатися VS Code на різних операційних системах, включаючи Windows, MacOS і Linux. Будь то ваша платформа, VS Code буде доступний і повністю функціональний на ній. Таким чином, ви не обмежені у виборі платформи для використання VS Code і можете насолоджуватися його перевагами, незалежно від операційної системи, яку віддаєте перевагу.

Emgu CV – це бібліотека для обробки зображень та КЗ мовою C#, яка заснована на бібліотеці з відкритим вихідним кодом OpenCV. OpenCV надає широкий спектр функцій для обробки зображень та КЗ, а Emgu CV дозволяє використовувати ці функції у програмах, написаних мовою C#.

Emgu CV підтримує кілька операційних систем, таких як Windows, Linux та Mac OS X. Крім того, бібліотека має чудову документацію та активну спільноту користувачів, які готові ділитися своїм досвідом та знаннями. Це робить Emgu CV доступною та корисною як для початківців, так і для досвідчених розробників [18, 36].

За допомогою Emgu CV можна реалізовувати багато завдань у галузі обробки зображень та КЗ, таких як:

– розпізнавання об’єктів;

– визначення кордонів та контурів;

– вилучення ознак зображень;

– розпізнавання осіб;

– сегментація зображень;

– детектування об’єктів, що рухаються;

– визначення відстані між об’єктами на зображенні;

– побудова 3D моделей на основі зображень.

Підсумовуючи, Emgu CV – це потужний інструмент обробки зображень і КЗ мовою C#. Ця бібліотека спрощує розробку програм, пов’язаних із КЗ, завдяки швидкому доступу до функцій OpenCV.

Open Source Computer Vision Library (OpenCV) – це бібліотека з відкритим вихідним кодом, призначена для створення програм, пов’язаних з обробкою зображень та КЗ. Бібліотека написана мовою C++ і може використовуватись на різних операційних системах, включаючи Windows, Linux, Mac OS, iOS та Android. Вона надає широкий спектр функцій для вирішення завдань КЗ, таких як розпізнавання об’єктів, вилучення ознак, сегментація зображень, розпізнавання облич, детектування об’єктів, що рухаються та інше.

OpenCV надає досить багато функціоналу для вирішення складних завдань у галузі КЗ, проте в деяких випадках його функціонал може виявитися недостатнім. У таких випадках можна використовувати компоненти OpenCV для створення власного рішення задачі майже будь-якої складності. Після

створення проекту можна проаналізувати його та виправити можливі недоліки. Рішення, отримане після створення проекту на основі компонентів бібліотеки, може бути використане надалі як зразок або шаблон для вирішення аналогічних завдань.

Найбільшою перевагою OpenCV є її висока продуктивність, яка забезпечується завдяки наступним факторам:

– OpenCV містить велику кількість оптимізованих алгоритмів, написаних мовою C++, які забезпечують швидке виконання завдань з обробки зображень та КЗ;

– можливість використовувати велику кількість потоків, що сприяє розпаралелюванню задач і підвищенню їх швидкості виконання на багатоядерних процесорах;

– OpenCV підтримує апаратне прискорення на певних платформах, включаючи NVIDIA CUDA та Intel OpenCL, що дозволяє значно прискорити виконання задач на графічних процесорах (GPU) та інших спеціалізованих обчислювальних пристроях;

– велика кількість оптимізованих алгоритмів для вирішення різних завдань КЗ та обробки зображень, включаючи розпізнавання об’єктів, сегментацію зображень, вилучення ознак і детектування об’єктів, що рухаються. Ці алгоритми забезпечують швидке та ефективне вирішення завдань;

– використання оптимізованих структури даних, такі як хеш-таблиці та дерева швидкого доступу, для ефективного доступу до великих обсягів даних та оптимізації роботи з пам’яттю.

Всі ці функції та можливості, у сукупності, перетворюють OpenCV на один із найбільш затребуваних та ефективних інструментів для обробки зображень та виконання завдань КЗ.

3.2 Особливості програмної реалізації

У процесі програмної реалізації були прийняті наступні кроки та реалізовані наступні рішення (рис. 3.1).

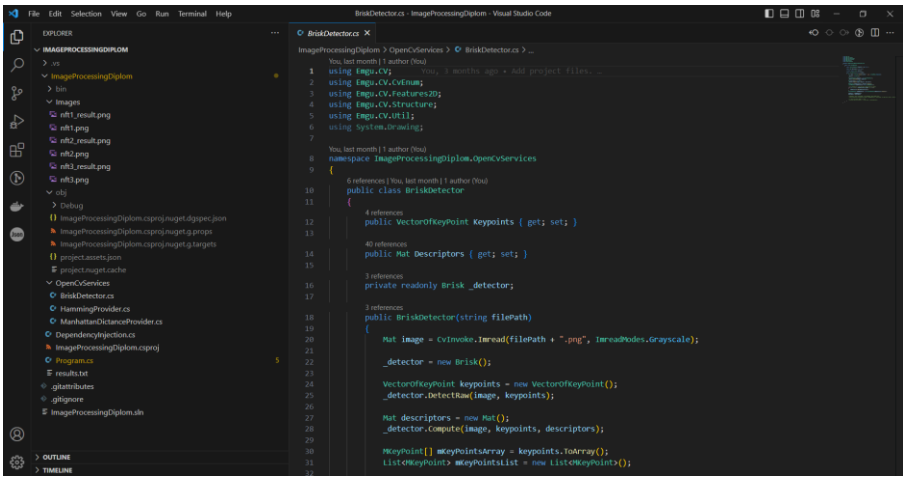

Рисунок 3.1 – Середовище розробки VS Code

Вхідний набір даних включав nft зображення (рис. 3.2) для яких детектор розпізнав КТ (рис. 3.3) та знайшов дескриптори зображень.

а) б) в)

Рисунок 3.2 – Вхідні зображення:

а) перший еталон; б) другий еталон; в) третій еталон

а) б) в)

Рисунок 3.3 – Зображення з координатами ключових точок:
а) перший еталон; б) другий еталон; в) третій еталон

Конструктор класу BriskDetector приймає шлях до файла зображення як параметр filePath. Потім він виконує такі дії:

– створюється об’єкт Mat image, у якому завантажується зображення з файлу, вказаного в filePath, з допомогою методу CvInvoke.Imread. Зображення читається у відтінках сірого (ImreadModes.Grayscale), щоб спростити обробку;

– створюється екземпляр класу Brisk за допомогою оператора New Brisk(). Цей клас надає функціональність для виявлення ключових точок та обчислення їх дескрипторів за допомогою алгоритму BRISK тобто створює детектор для конкретного зображення;

– створюється об’єкт VectorOfKeyPoint keypoints, який міститиме виявлені ключові точки на зображенні;

– метод detector.DetectRaw(image, keypoints) викликається виявлення не відфільтворованих ключових точок на зображенні. Результат виявлення зберігається в ключових точках;

– створюється об’єкт Mat descriptors, який міститиме обчислені дескриптори для ключових точок;

– метод detector.Compute(image, keypoints, descriptors) викликається обчислення дескрипторів для виявлених ключових точок. Результат обчислень зберігається;

– створюється масив mKeyPointsArray, у якому зберігаються ключові точки з keypoints як масиву;

– цикл for проходить масивом mKeyPointsArray і відбирає лише значуші дескриптори для того щоб зменшити їх кількість до 500;

– створюється новий об’єкт VectorOfKeyPoint top500KeyPoints, який копіює ключові точки з mKeyPointsList. Він міститиме максимум 500 ключових точок.

Значення top500KeyPoints надається властивості Keypoints класу BriskDetector, а значення descriptors присвоюється властивості Descriptors.

В результаті виконання конструктора BriskDetector, об’єкт даного класу міститиме інформацію про ключові точки та їх дескриптори для зазначеного зображення, що дозволяє використовувати їх для подальшої обробки або аналізу зображень.

Клас DictanceProvider містить методи для роботи з відстанями та обчисленням мінімальних значень у двовимірних масивах.

У методі CountThresholdMathes виконується підрахунок кількості елементів у масиві відстаней, які менші від заданого порогового значення. Для цього створюється масив minimums, потім відбувається ітерація елементів масиву, і якщо значення елемента менше порогового значення, то лічильник count збільшується. Наприкінці метод повертає значення кількості голосів які відносить до конкретного еталону.

Метод FindMinimumsOfDistances обчислює мінімальні значення кожного рядка масиву в матриці відстаней. Створюється масив мінімумів з тією самою кількістю елементів, що й число рядків у матриці. Потім відбувається подвійна ітерація елементів матриці distances, де кожен елемент порівнюється з поточним мінімальним значенням відповідного рядка масиву мінімумів. Якщо значення елемента менше поточного мінімального значення, воно стає новим мінімальним значенням. Наприкінці метод повертає масив minimums.

Таким чином, клас DictanceProvider надає зручні методи для роботи з відстанями та обчислення мінімальних значень у двовимірних масивах, що може бути корисним при обробці даних, пов’язаних із зображеннями.

Клас HammingProvider, який надає методи для роботи з алгоритмом Хеммінга та обчислення відстаней. Структура VoteResult містить список результатів, призначений для збереження результатів голосування.

Клас HammingProvider має кілька константних полів, включаючи кількість дескрипторів, робміри дескриптора, поріг, які використовуються у методах класу.

Метод FindUniqueMathesForAllEtalon приймає список матриць еталонів і повертає список двовимірних масивів результат, де кожен масив містить значення унікальних збігів для відповідної матриці etalons. Для цього метод виконує ітерацію кожної матриці і викликає метод знаходження унікального значення, передаючи поточну матрицю і список еталонів. Результати додаються до списка результату, який потім повертається.

Метод FindUniqueValue приймає матрицю еталону для порівняння і список матриць і повертає двовимірний масив результат, що містить значення унікальних збігів між ними. Метод робить подвійну ітерацію за елементами результат, де кожен елемент отримує значення повторів, що викликаються для відповідної пари матриць.

Метод FindCountOfRepeadByHammingLenght приймає байтовий масив descriptor та матрицю etalon і повертає кількість повторень, ґрунтуючись на довжині відстані Хеммінгу між descriptor та кожним елементом матриці etalon. Метод виконує ітерацію за елементами descriptor та викликає метод FindHammingLenghtForDescriptors для обчислення відстані Хеммінгу між елементами descriptor та відповідними елементами матриці etalon. Якщо відстань менша від порогового значення THREADHOLD_DESC, лічильник result збільшується. Потім метод повертає значення результату.

Методи FindHammingLenghtForDescriptors, FindHammingDistance, FindHammingDistanceWithIndexes і FindMinHammingLenght виконують

обчислення відстаней Хеммінга між двома байтовими масивами дескрипторів або матрицями дескрипторів, використовуючи подвійні ітерації по матрицях.

Метод VoteEtalon приймає список матриць etalons та матрицю descriptors. Він виконує голосування кожного дескриптора з матриці descriptors, визначаючи найбільш підходящу матрицю etalon.

Метод починає ітерацію кожного дескриптора з матриці descriptors. Для кожного дескриптора він виконує ітерацію за списком etalons, порівнюючи його з кожною матрицею etalon. Для порівняння використовується метод FindMinHammingLenght, який обчислює відстань Хеммінгу між поточним дескриптором та кожним дескриптором із матриці etalon.

Якщо відстань між дескрипторами менша за значення minDistance, яке спочатку встановлено рівним DESC_SIZE, то minDistance оновлюється, а також перевіряється, якщо minDistance менше порогового значення THREADHOLD_DESC. Якщо умова виконується, то індекс відповідної матриці etalon записується в змінну indexOfEtalonToVote.

Після закінчення внутрішньої ітерації за списком еталонів перевіряється значення indexOfEtalonToVote. Якщо вона не дорівнює -1, це означає, що була знайдена відповідна матриця etalon для поточного дескриптора. У такому разі значення indexOfEtalonToVote використовується для інкрементації відповідного елемента списку результатів.

Після завершення зовнішньої ітерації за дескрипторами, структура VoteResult створюється та ініціалізується. Потім для кожної матриці etalon результат голосування записується у відповідний елемент списку результатів. Підсумковий результат повертається із методу.

Даний метод дозволяє визначити найбільш підходящу матрицю etalon для кожного дескриптора з матриці descriptors з урахуванням відстані Хеммінга. Це може бути використане, наприклад, у завданнях розпізнавання образів або пошуку відповідності між зображеннями.

Файл DependencyInjection.cs містить статичний клас DependencyInjection, у якому визначено метод AddServices. Цей метод

розширює функціональність класу IServiceCollection, який є частиною фреймворку Microsoft.Extensions.DependencyInjection і надає механізм реєстрації та дозволу залежностей.

Усередині методу AddServices відбувається реєстрація двох сервісів: HammingProvider та ManhattanDictanceProvider. Ці послуги додаються до колекції послуг за допомогою методу AddTransient. Метод AddTransient вказує, що кожного разу, коли запит цих сервісів повинен створюватися новий екземпляр. Таким чином, при кожному запиті HammingProvider та ManhattanDictanceProvider буде створюватися новий об’єкт.

Після реєстрації сервісів метод AddServices повертає об’єкт services. Це дозволяє використовувати ланцюжок дзвінків для додавання інших сервісів або налаштування залежностей в інших частинах коду.

Цей файл є частиною конфігурації залежностей програми та полегшує керування та дозвіл залежностей в інших частинах програми (рис. 3.4).

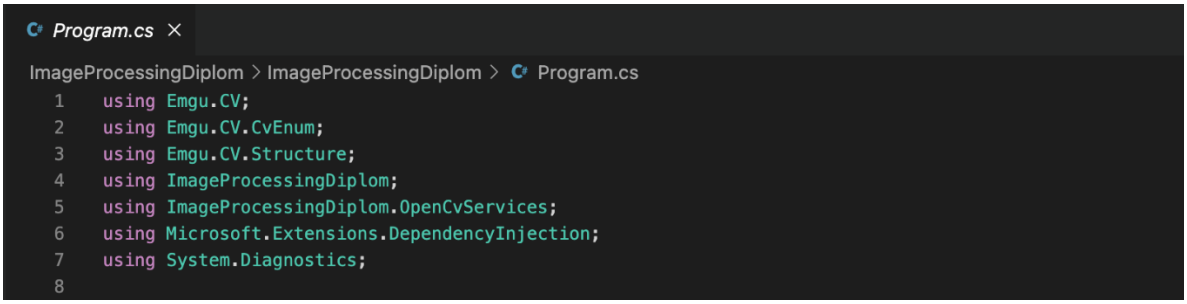

Рисунок 3.4 – Бібліотеки, які підключаються до файлу Program.cs

У файлі Program.cs використовується бібліотека Emgu.CV для обробки зображень та виконує деякі операції з обробки дескрипторів зображень.

Спочатку відбувається налаштування залежностей за допомогою Dependency Injection. Створюється контейнер служб і додаються служби HammingProvider та ManhattanDictanceProvider з ImageProcessingDiplom.OpenCvServices.

Потім визначаються шляхи до зображень у змінній imagePathes. Далі створюються екземпляри класу BriskDetector для кожного зображення та

витягуються їх дескриптори. Наступним кроком виводиться кількість знайдених ключових точок на кожному зображенні.

Запускається секундомір і викликається метод FindUniqueMathesForAllEtalon для обчислення унікальних збігів між дескрипторами. Результат зберігається в змінну uniqueNumbers. Секундомір зупиняється і виводиться час, витрачений на обчислення.

Далі слідує блок коду, який створює і заповнює двовимірний список b з розмірами 3x500. Потім виконується вкладений цикл, в якому значенням b додаються суми відповідних елементів з uniqueNumbers, виключаючи значення з поточного зображення. Далі у двовимірний список s копіюються значення з uniqueNumbers, а потім кожне значення поділяється на 1500.

Запускається секундомір stopwatch3, і кожної пари зображень викликається метод FindHammingDistance, щоб обчислити відстань Хеммінга між дескрипторами. Потім секундомір зупиняється і виводиться час, витрачений на обчислення. Для кожної пари зображень викликається метод CountThresholdMathes, щоб підрахувати граничну кількість збігів, ґрунтуючись на відстанях Хеммінга.

Потім створюються індекси для вибору найбільш відповідних елементів b. Запускається секундомір stopwatch4, і кожної пари зображень викликається метод FindHammingDistanceWithIndexes, щоб обчислити відстань Хеммінга з допомогою вибраних індексів. Потім секундомір зупиняється і виводиться час, витрачений на обчислення.

Наприкінці виводяться результати збігів між парами зображень. Вимірювання часу зупиняється і виводиться в консоль. Обчислюється кількість збігів кожної пари дескрипторів з допомогою методу CountThresholdMathes, результати виводяться в консоль і додаються змінну results. Результати виводяться у текстовий файл results.txt.

Ця ділянка коду виконує виявлення ключових точок на зображеннях, обчислення дескрипторів, обчислення унікальності та збігу між дескрипторами, а також записує результати у файл.

3.3 Інструкція користувача

Для запуску програми необхідно скачати редистрибутив .NET 6 для відтворення застосунку з офіційно сайту Microsoft, в залежності від типи системи на якій буде відтворюватись необхідно вибрати підходящий тип архітектури x86 або x64 та вид операційної системи.

Далі необхідно скопіювати зображення в форматі png в папку Images всередині проекту та змінити їх назву на image1, image2 і так далі в залежності від кількості еталонів які оцінюються.

Далі необхідно в тій самій папці, де знаходиться проект з розширеням sln, запустити термінал та прописати команду dotnet run (рис. 3.5).

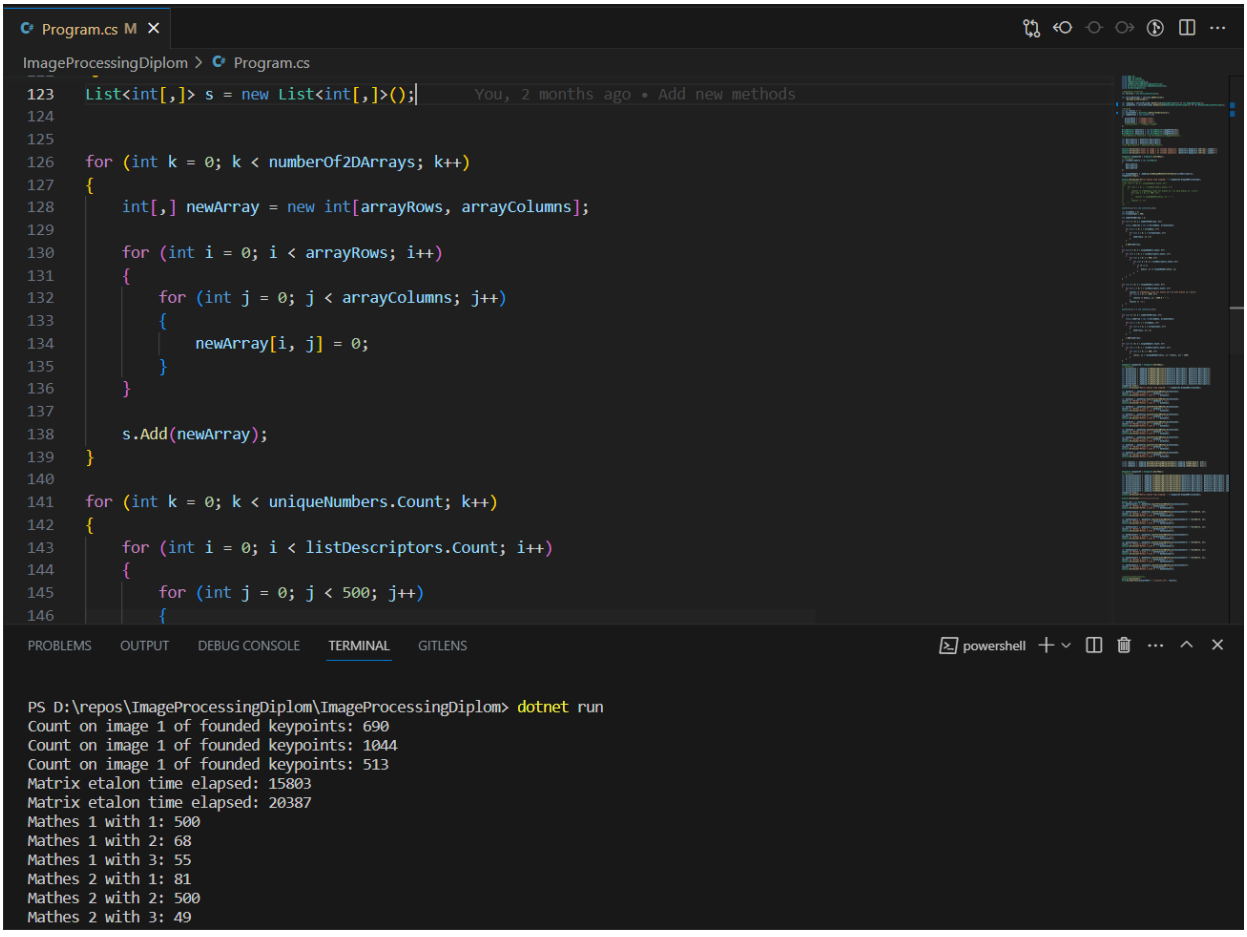

Рисунок 3.5 – Запуск проекту в середі розробки Visual Studio Code

3.4 Аналіз експериментальних результатів

В ході використання методу на основі параметру унікальності та віднесення їх до певного класу, вдалося оцінити час та результати голосувань дескрипторів для різних методів. Для класичного методу голосування було обрано відповідно по 500 дескрипторів, обраховано хемінгову відстань опису зображення від всіх еталонів (табл. 3.1). Як бачимо, класифікація проведена правильно і програма витратила близько 21391 мс.

Таблиця 3.1 – Класичний метод голосування на 500 дескрипторів

Еталони	1	2	3
1	500	68	55
2	81	500	49
3	58	52	500
Обсяг часу		21391 мс	

Для методу оцінки з відбором дескрипторів за показником інформативності альфа було отримано приблизно такої ж якості результати класифікації (табл. 3.2). Кількість голосів співвідносна з класичним методом навіть для еталонів, до яких зображення не було віднесене. Отже, цей метод має достатню точність, при цьому час виконання зменшився до 4786 мс.

Таблиця 3.2 – Метод голосування для 100 дескрипторів з найбільшою інформативністю за показником альфа

Еталони	1	2	3
1	100	8	7
2	7	100	6
3	12	10	100
Обсяг часу		4786 мс	

Отже, відбір лише частини дескрипторів за критерієм інформативнсті альфа є доцільним і може застосовуватись для суттєвого пришвидщення процесу класифікації, адже обробка даних пов’язана в визначенням показників інформативності еталонів. Вона проводиться на етапі попередньої обробки і її час на виконання класифікації не враховується.

Подібні результати кажуть про необхідність перевірки результативності цього методу на більшому наборі вхідних даних для більш точного визначення виграшу в часі в порівнянні з класичним методом, або для вдосконалення цього методу.

Для методу оцінки інформативності за показником бета порівняння проводиться за показником, що вказує на неподібність опису до інших еталонів, тому результат класифікації вказує на більшу відповідність тільки найменших відстаней при класифікації і не враховує відалення до інших еталонів. Результати класифікації (табл. 3.3) такж виявились ефективними і швидшими за класичний метод і програма виконалась за 3824 мс.

Таблиця 3.3 – Метод голосування для 100 дескрипторів з найбільшою інформативністю за показником бета

Еталони	1	2	3
1	100	6	75
2	5	100	2
3	6	3	100
Витрачено часу		3284 мс	

Через більш чітке віднесення до конкретного класу, ефективність оцінки за показником бета буде зростати для схожих зображень, адже навіть при великій кількості дескрипторів, які матимуть невелику хемінгову відстань між собою, для класифікації будуть відбиратись показники з найбільшим ступенем інформативності на основі інших класів.

Загалом результати показали що методи класифікації з використанням показників інформативнсті для компонентів опису є перспективним напрямом для дослідження та мають потенціал для використання замість традиційного метода голосування.

ВИСНОВКИ

У рамках кваліфікаційної роботи був розроблений і реалізований метод класифікації зображень за допомогою оцінки значень інформативності дескрипторів зображень.

На основі редукції кількості ознак в складах опису, тобто зменшення кількості необхідних операцій, виконується відбір дескрипторів за значенням інформативності. Це дозволяє суттєво збільшити швидкість виконання класифікації за рахунок проведення попереднього аналізу складу дескрипторів для бази еталонів.

Практичні рекомендації із проведеного дослідження полягають у результативному використанні методів попереднього аналізу даних у класифікаторах, які включать відбір підмножини опису за певними показниками..

Практична значущість роботи – досягнення значного пришвидшення часу, витраченого на процес класифікації за рахунок результативного відбору числа дескрипторів з найбільшими показниками інформативності. Це допоможе для подальшої інтеграції обговорюваних методів у системах КЗ.

Перспективи роботи можуть бути пов’язані з дослідженням розробленого методу на більшій вибірці даних та для зображень з високою кількістю ключових точок.

Результати роботи апробовано у вигляді тез доповіді під час XXVII Міжнародного молодіжного форуму «РАДІОЕЛЕКТРОНІКА І МОЛОДЬ У XXI СТОЛІТТІ» [30].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Гороховатський, В. О., & Творошенко, І. С. (2022). Аналіз багатовимірних даних за описом у формі множини компонент.

2. Гороховатський, В.А., Путятін, Е.П., & Столяров, В.С. (2017). Дослідження результативності структурних методів класифікації зображень із застосуванням кластерної моделі даних. Радіоелектроніка, інформатика, управління.

3. Верес, О. М., Кісь, Я. П., Кугівчак, В. А., & Рішняк, І. В. (2018). Вибір методів для пошуку однакових або схожих зображень. *Вісник Національного університету "Львівська політехніка". Серія: Інформаційні системи та мережі*, (887), 43-50.

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

5. Linda, G. (2001). Shapiro george c. stockman. computer vision.

6. Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). Brief: Binary robust independent elementary features. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11* (pp. 778-792). Springer Berlin Heidelberg.

7. 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.

8. Гороховатський, В. О., & Гадецька, С. В. (2020). Статистичне оброблення та аналіз даних у структурних методах класифікації зображень.

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

10. Gadetska, S. V., & Gorokhovatskyi, V. O. (2018). Statistical measures for computation of the image relevance of visual objects in the structural image classification methods. *Telecommunications and Radio Engineering*, *77*(12).

11. Tuytelaars, T., & Mikolajczyk, K. (2008). Local invariant feature detectors: a survey. *Foundations and trends® in computer graphics and vision*, *3*(3), 177-280.

12. Гороховатський, В. О., Руденко, Д. О., & Сірик, Т. О. (2019). Дослідження системи ієрархічних ознак при блочному поданні опису у складі множини ключових точок зображення.

13. Introduction to SIFT (Scale Invariant Feature Transform). URL: https://medium.com/data-breach/introduction-to-sift-scale-invariant-feature-transform-65d7f3a72d40 (дата звернення 14.04.2023).

14. 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.

15. Introduction to FAST (Features from Accelerated Segment Test). URL: https://medium.com/data-breach/introduction-to-fast-features-from-accelerated-segment-test-4ed33dde6d65 (дата звернення 14.04.2023).

16. Introduction to BRIEF (Binary Robust Independent Elementary Features). URL: https://medium.com/data-breach/introduction-to-brief-binary-robust-independent-elementary-features-436f4a31a0e6 (дата звернення 21.04.2023).

17. BRISK. URL:https://cvexplained.wordpress.com/2020/07/28/10-10-9-brisk/ (дата звернення 22.04.2023).

18. 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.

19. Гороховатський В., Передрій О., Творошенко І., Марков Т. (2023) Матриця відстаней для множини компонентів структурного опису як інструмент для створення класифікатора зображень, *Сучасні інформаційні системи*, 7(1), С. 5-13.

20. Гороховатський, В.А., & Путятін, Е.П (2008). Структурне розпізнавання зображень на основі моделей голосування ознак характерних точок. Реєстрація, зберігання і обробка даних.

21. Gorokhovatskyi V.A. (2018) Image Classification Methods in the Space of Descriptions in the Form of a Set of the Key Point Descriptors. Telecommunications and Radio Engineering, 77 (9), pp. 787-797.

22. Gorokhovatsky, V.O. and Gadetska, S.V. (2019) Determination of Relevance of Visual Object Images by Application of Statistical Analysis of Regarding Fragment Representation of their Descriptions, Telecommunications and Radio Engineering, 78 (3), pp. 211–220.

23. Коцовський, В. М. (2014). Теорія розпізнавання образів. Лекції.

24. Математичні принципи масштабно-інваріантних дескрипторів зображень. URL: http://pzs.dstu.dp.ua/ComputerGraphics/d&d/5.html / (дата звернення 26.04.2023).

25. K-Nearest Neighbor(KNN) Algorithm for Machine Learning. URL: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning/ (дата звернення 30.04.2023).

26. Daradkeh, Y. I., Tvoroshenko, I., Gorokhovatskyi, V., Latiff, L. A., & Ahmad, N. (2021). Development of effective methods for structural image recognition using the principles of data granulation and apparatus of fuzzy logic. *IEEE Access*, *9*, 13417-13428.

27. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., and Zeghid M. (2022) Tools for fast metric data search in structural methods for image classification, *IEEE Access*, 10, pp. 124738-124746.

28. Gorokhovatskyi V., Tvoroshenko I., Kobylin O., and Vlasenko N. (2023) Search for visual objects by request in the form of a cluster representation for the structural image description, *Advances in Electrical and Electronic Engineering*, 21(1), pp. 19-27.

29. 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.

30. Gorokhovatskyi V.A., Zamula A.A. (2016) Employment of Intelligent Technologies in Multiparametric Control Systems. Telecommunications and Radio Engineering. Vol. 75, No 19, p. 1775–1785.

31. Gorokhovatskiy, V. A. (2016). Efficient Estimation of Visual Object Relevance during Recognition through their Vector Descriptions. *Telecommunications and Radio Engineering*, *75*(14).

32. Gorokhovatskiy, V. A. (2011). Compression of descriptions in the structural image recognition. *Telecommunications and Radio Engineering*, *70*(15).

33. Гороховатський В.О., Передрий Е.О. (2009) Кореляційні методи розпізнавання зображень шляхом голосування систем фрагментів. Радіоелектроніка, інформатика, управління, №1 (20), с.74–81.

34. Gorokhovatskyi, V., Stiahlyk, N., Tsarevska, V. (2021). Combination method of accelerated metric data search in image classification problems. Advanced Information Systems, 5 (3), pp. 5–12.

35. 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. Радіоелектроніка, інформатика, управління, №4 , с. 85–94.

36. Gorokhovatskyi V., Gadetska S., Ponomarenko R. (2020) Recognition of Visual Objects Based on Statistical Distributions for Blocks of Structural Description of Image. Proc. of the XV Int. Scientific Conference “Intellectual Systems of Decision Making and Problems of Computational Intelligence” (ISDMCI’2019), Ukraine, May 21–25, 2019, pp. 501-512.

37. Gadetska, S. V., Gorokhovatskyi, V. O., Stiahlyk, N. I., & Vlasenko, N. V. (2021). Statistical data analysis tools in image classification methods based on the description as a set of binary descriptors of key points. *Radio Electronics, Computer Science, Control*, (4), 58-68.

38. Gadetska S., Gorokhovatskyi V., Stiahlyk N., Vlasenko N. (2022) Aggregate Parametric Representation of Image Structural Description in Statistical

Classification Methods. In CEUR Workshop Proceedings: Computer Modeling and Intelligent Systems (CMIS-2022), 3137, pp. 68-77.

39. Gorokhovatskyi, O., Peredrii, O., Gorokhovatskyi, V., Vlasenko, N. (2023) Explanation of CNN Image Classifiers with Hiding Parts. In: J. Benois-Pineau, R. Bourqui, D. Petkovic, G. Quenot (eds), Explainable Deep Learning Artificial Intelligence, pp. 125-146, Academic Press, 346 p.

40. Gorokhovatskyi, V., Vlasenko, N. (2021). Редукція опису зображення у складі множини дескрипторів на основі метричного критерію інформативності. Advanced Information Systems, 5(4), pp. 10-16.

41. Гороховатський, В. О., Гадецька, С. В., & Пономаренко, Р. П. (2019). Статистичні розподіли та ланцюжкове подання даних при визначенні релевантності структурних описів візуальних об’єктів.

42. Гороховатський, В. О., Творошенко, І. С., & Сидоренко, Д. (2021). Класифікація зображень із використанням кластерного подання.

43. Ibrahim, D. Y., Gorokhovatskyi, V., Tvoroshenko, I., & Zeghid, M. (2022). Cluster representation of the structural description of images for effective classification.

44. Gorokhovatskyi O., Gorokhovatskyi V., Peredrii O. (2018) Analysis of Application of Cluster Descriptions in Space of Characteristic Image Features. Data, 3(4), 52.

45. Чмутов, Ю. В. (2023). Дослідження ортогональної трансформації описів зображень для формування класифікаційних ознак.

46. Шевляков, А. С. 2023 Моделювання класифікатора зображень на основі критерію унікальності ознак. 27-ий міжнародний молодіжний форум «РАДІОЕЛЕКТРОНІКА І МОЛОДЬ У XXI СТОЛІТТІ», т. 7, С. 20-21