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)

РОЗРОБЛЕННЯ МЕТОДУ СТРУКТУРНОЇ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ З ВИКОРИСТАННЯМ КВАНТУВАННЯ У ПРОСТОРІ ОЗНАК
(тема)

Виконав:
здобувач 4 року навчання,
групи ІТІНФ-21-3
Щербак Д. Д.
(прізвище, ініціали)

Спеціальність 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

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

2025 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту

Кафедра Інформатики

Рівень вищої освіти перший (бакалаврський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Щербак Дар'ї Дмитрівні
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення методу структурної класифікації зображень з використанням квантування у просторі ознак

затверджена наказом університету від 19 травня 2025 року № 381Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 26 травня 2025 р.
3. Вихідні дані до роботи науково-методична та науково-технічна література, матеріали конференцій, дані інтернет-мережі, бібліотека комп'ютерного зору з відкритим кодом OpenCV.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд традиційних методів класифікації зображень та вивчення методів з використанням апарату ключових точок та дескрипторів.
2. Розробка методу структурної класифікації на основі квантування ознак та метрики Танімото.
3. Програмне моделювання та тестування модифікованого методу.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність задачі класифікації зображень, постановка задачі кваліфікаційної роботи, еталони і зображення поза базою, метод класифікації зображень методом голосування, схема трансформації опису зображень за результатом квантування, аналіз результатів програмного моделювання.

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

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	07.04.2025	
2	Аналіз завдання, підбір літератури	09.04.25-12.04.25	
3	Аналіз літератури з досліджуваної проблеми	13.04.25-15.04.25	
4	Аналіз технічних засобів	16.04.25-22.04.25	
5	Розробка методу	23.04.25-27.04.25	
6	Програмна реалізація	28.04.25-11.05.25	
7	Оформлення пояснювальної записки	12.05.25-20.05.25	
8	Перевірка на нормоконтроль	21.05.25-01.06.25	
9	Перевірка на плагіат	21.05.25-01.06.25	
10	Рецензування	21.05.25-01.06.25	
11	Підготовка презентації та доповіді	21.05.25-18.06.25	
12	Занесення роботи в електронний архів	02.06.25-18.06.25	
13	Попередній захист кваліфікаційної роботи	02.06.25-18.06.25	

Дата видачі завдання 7 квітня 2025 р.

Здобувач _____
(підпис)

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

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 71 с., 7 табл., 21 рис., 44 джерела.

КЛАСИФІКАЦІЯ ЗОБРАЖЕНЬ, СТРУКТУРНІ МЕТОДИ, ДЕСКРИПТОРИ BRISK, КВАНТУВАННЯ ОЗНАК, МЕТРИКА ТАНІМОТО, ШВИДКОДІЯ, ТОЧНІСТЬ.

Об’єктом роботи є методи класифікації зображень за множиною дескрипторів ключових точок.

Метою роботи є розроблення модифікації для методу структурної класифікації зображень на основі квантування у просторі ознак, що передбачає застосування бінарних дескрипторів BRISK, методу *k*-середніх та метрики Танімото для швидкісного оцінювання релевантності об’єктів.

Застосовано методи аналізу багатовимірних даних, кластеризацію та програмне моделювання класифікаторів з використанням голосування й метрики Танімото. Запропоновано порогову фільтрацію множини дескрипторів для усунення хибних даних.

Проведено тестування розроблених методів і оцінку їх точності та швидкодії. Розроблений метод суттєво підвищує швидкодію класифікації зображень зі збереженням високої точності.

IMAGE CLASSIFICATION, STRUCTURAL METHODS, BRISK DESCRIPTORS, FEATURE QUANTIZATION, TANIMOTO METRIC, SPEED, ACCURACY.

The object of the work is image classification methods based on sets of keypoint descriptors.

The aim of the work is is to develop a modification of the structural image classification method based on feature space quantization, involving the use of binary BRISK descriptors, the *k*-means method, and the Tanimoto metric for fast relevance estimation of objects.

Methods of multidimensional data analysis, clustering, and software modeling of classifiers using voting and the Tanimoto metric were applied. Threshold-based filtering of the descriptor set was proposed to eliminate erroneous data.

The developed methods were tested, and their accuracy and processing speed were evaluated. The proposed method significantly improves image classification speed while maintaining high accuracy.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

КТ – ключові точки зображення

BRISK – Binary Robust Invariant Scalable Keypoints

FAST – Features from Accelerated Segment Test

BRIEF – Binary Robust Independent Elementary Features

SIFT – Scale-invariant Feature Transform

SURF – Speeded-Up Robust Features

ORB – Oriented FAST and Rotated BRIEF

AKAZE – Accelerated-KAZE

DAISY – Dense Descriptor Similarity

OpenCV – Open-Source Computer Vision Library

BoVW – модель bag-of-visual-words model

t-SNE – t-Distributed Stochastic Neighbor Embedding

ВСТУП

У сфері комп’ютерного зору та системах, що у ній розробляються, часто постає задача створення методів класифікації зображень, де кожен окремий клас представляється у вигляді множин дескрипторів ключових точок. У деяких з таких систем пріоритет ставиться на параметр швидкодії виконання обчислень. Такими системами зазвичай є системи, що виконують завдання комп’ютерного зору у реальному часі.

Важливими критеріями при виборі алгоритму класифікації зображень є швидкодія та точність класифікатора. Одним із напрямів підвищення ефективності є використання структурних методів, де опис об’єктів подається множиною дескрипторів ключових точок, що дозволяє уникнути обмежень, притаманних сучасним нейронним мережам, таких як висока розмірність, потреба у значних обсягах пам’яті, або складність досягнення інваріантності до афінних перетворень. У структурних методах ключовою задачею є визначення схожості чи відмінності для двох множин багатовимірних векторів, що описують візуальні об’єкти. Задля забезпечення високих показників класифікації зображень при зіставленні таких множин кожен вектор даних розглядається разом із деяким околом, що визначає його еквівалентність у багатовимірному просторі.

Як метод апроксимації для чисельної множини багатовимірних векторів опису зображень задля підвищення швидкодії можна використовувати кластеризацію. Одним із підходів до цього є проєктування простору даних у нову квантовану форму – наприклад, шляхом кластеризації чи хешування. У такій моделі кожен вектор ідентифікується зі своїм центром кластеру, що дозволяє вважати дані в межах одного кластеру еквівалентними. Це спрощує процес класифікації, дозволяє уникнути витрат на пошук найближчого сусіда та відкриває можливості застосування математично обґрунтованих метрик подібності.

Актуальність теми бакалаврської роботи полягає у важливості створення швидкодіючих алгоритмів класифікації зображень для використання комп’ютерними системами, що потребують ефективної обробки візуальної інформації у реальному часі.

Практична значущість бакалаврської роботи складає розробка та дослідження способу модифікації простору дескрипторів ключових точок зображень шляхом квантування, з подальшою класифікацією отриманого опису у перетвореному просторі з використанням математичних метрик для оцінки релевантності з еталонними описами.

Завданням роботи є проєктування та реалізація методу класифікації зображень на основі попереднього перетворення множини дескрипторів у квантовану форму із подальшим оцінюванням результативності такого підходу щодо швидкодії та точності.

1 АНАЛІЗ ЗОБРАЖЕНЬ У КОМП'ЮТЕРНОМУ ЗОРІ

1.1 Загальні засади комп'ютерного зору

Комп'ютерний зір – це надзвичайно складна галузь, яка займається вирішенням широкого спектру задач і дає можливість комп'ютерам аналізувати, інтерпретувати та розуміти візуальні дані, такі як зображення та відео [1]. Комп'ютери «бачать» світ не так, як ми: якщо для людини зір – це найважливіший орган відчуття, то для машини зображення – це просто набір чисел, що показують інтенсивність пікселів (рис. 1.1). Тож головна мета комп'ютерного бачення – наблизити машинне сприйняття за результативністю опрацювання до людського шляхом інтерпретації цих чисел [2].

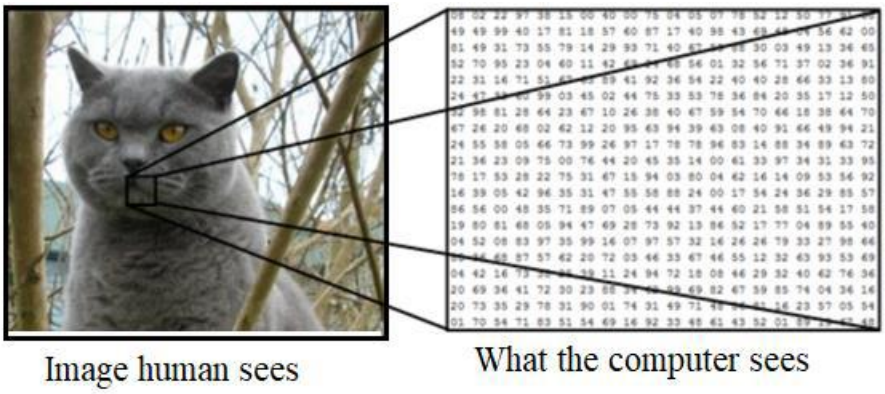


Рисунок 1.1 – Сприйняття об'єктів людиною і машиною

Зображення у цифровому вигляді представляється як матриця пікселів, де кожен піксель має певне числове значення, яке відображає його яскравість або колір. Найчастіше в задачах комп'ютерного зору використовуються зображення у градаціях сірого (або напівтонові зображення), де кожен піксель має одне значення яскравості – зазвичай у межах від 0 до 255, де 0 відповідає чорному кольору, 255 – білому, а проміжні значення – відтінкам сірого. Така форма подання дозволяє зменшити обчислювальну складність, що особливо важливо при роботі з великими обсягами даних або в реальному часі [3].

Галузь комп’ютерного зору виникла на стику інформатики, математики, нейронаук та інженерії. Вона тісно пов’язана з машинним навчанням, обробкою зображень, робототехнікою, адже має на меті моделювати людське зорове сприйняття й витягувати змістовну інформацію з візуального середовища [4].

Історія комп’ютерного зору починається ще з 1960-х років, коли здійснювались перші спроби автоматичної обробки зображень з супутників і виявляти об’єкти у просторі. У 1980-х роках розвиток нейробіології став поштовхом для створення перших алгоритмів машинного навчання для розпізнавання зображень. Однак справжній стрибок стався у 2000-х, коли з’явилися глибокі нейронні мережі та згорткові моделі, які суттєво підвищили точність обробки зображень [5].

Одним із сучасних прикладів практичного застосування технологій комп’ютерного зору є система моніторингу соціальної дистанції під час пандемії COVID-19, запропонована дослідницькою групою Artificial Intelligence for Media and Humanities (AIMH). Ця вбудована система реального використання дозволяє виявляти пішоходів, відстежувати їхнє переміщення, оцінювати скупчення людей і контролювати дотримання правил соціального дистанціювання. Система базується на модульній візуальній обробці зображень і забезпечує високий рівень точності в реальних умовах. Її ефективність було продемонстровано під час тестування на об’єкті в Італії у період дії карантинних обмежень (рис. 1.2). Цей приклад ілюструє лише одну з численних сфер, де комп’ютерний зір відіграє ключову роль у забезпеченні безпеки, автоматизації та прийняття рішень у режимі реального часу [6].

Сьогодні багато напрямків використовують штучний інтелект і, зокрема, комп’ютерний зір для оптимізації та автоматизації процесів, тож ця галузь продовжує активно розвиватись. Охорона здоров’я, сільське господарство, автономні транспортні засоби, системи безпеки та відеоспостереження, віртуальна та доповнена реальність – ось лише невеликий список сфер, в яких застосовується комп’ютерне бачення [1, 2, 4].

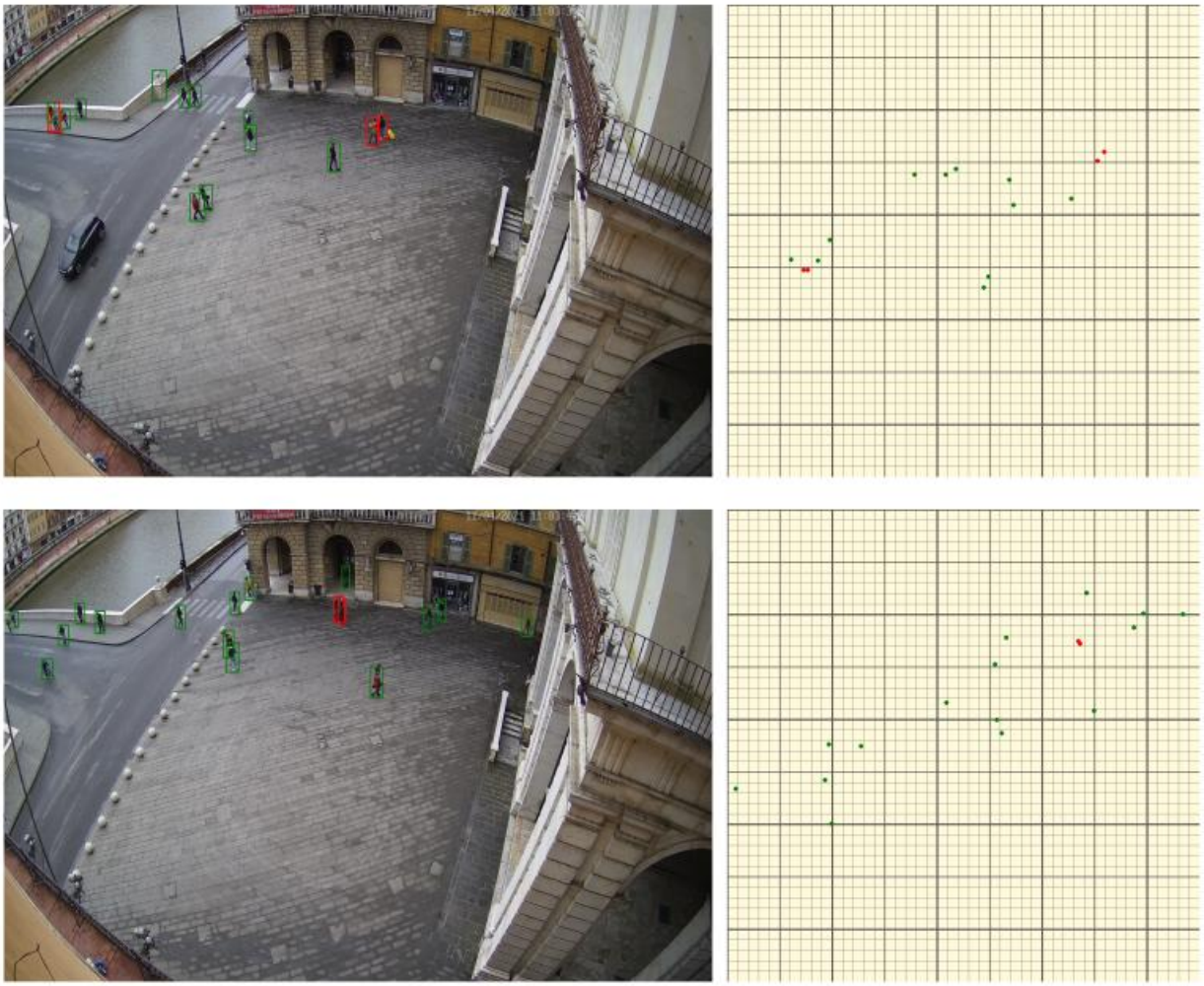


Рисунок 1.2 – Приклади роботи модуля вимірювання соціальної дистанції

Серед найпоширеніших задач, які охоплює комп’ютерний зір, є:

– виявлення об’єктів (object detection), що передбачає одночасне виявлення і класифікацію об’єктів на зображенні;

– класифікація зображень (image classification) – задача присвоєння зображенню одного або кількох класів на основі його опису;

– виділення ознак (feature extraction), що означає формування представлення зображення у вигляді ознак, що зберігають найважливішу інформацію про нього;

– відстеження руху (motion tracking) дозволяє відслідковувати рух об’єктів на відео в часі;

– сегментація зображень (image segmentation) полягає у розбитті зображення на області, що мають спільні характеристики [4].

На рисунку 1.3 зображено основні напрями комп’ютерного зору разом із прикладами їх практичного застосування, таких як виявлення пішоходів, класифікація товарів, відстеження відстані до об’єктів, або сегментація відео на значущі фрагменти тощо [4].

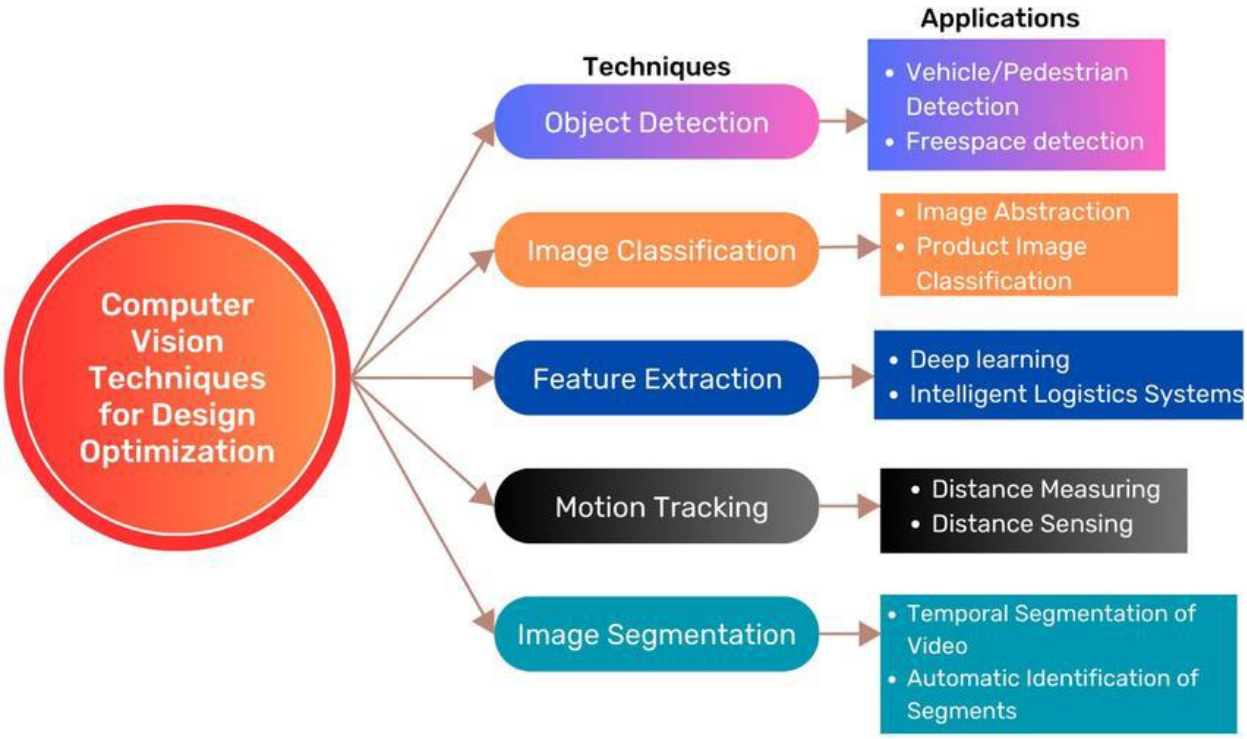


Рисунок 1.3 – Напрямки у комп’ютерному зорі

Отже, комп’ютерний зір охоплює широкий спектр задач і знаходить застосування у найрізноманітніших галузях. Ефективність вирішення цих задач значною мірою залежить від обраного підходу до аналізу зображень, який визначається специфікою предметної області та вимогами до точності й швидкодії системи.

1.2 Структурний аналіз зображень за множиною дескрипторів

Різні прикладні задачі потребують різноманіття підходів до їх вирішення, основними серед яких є: на основі нечіткої логіки; на базі

згорткових нейронних мереж та глибокого навчання; з використанням детекторів і дескрипторів [7].

Підхід на основі нечіткої логіки орієнтований на роботу в умовах невизначеності. Теорія нечіткої логіки дозволяє гнучко інтерпретувати зображення, змінюючи рівень узагальнення та застосовуючи правила, подібні до людського досвіду [7].

Підхід на базі згорткових нейронних мереж є домінуючим у сучасних системах глибокого навчання. Він побудований на використанні тривимірних шарів з локальними зв’язками для виділення ознак із вхідних даних, таких як ширина, висота та колірні канали зображення [7, 8].

Підхід з використанням детекторів і дескрипторів заснований на виявленні особливих точок зображення та їх числовому описі. Цей метод дозволяє досягати точних результатів навіть при менших обчислювальних витратах [7, 9-24].

Саме цей підхід і буде основою вивчення в даній роботі, адже він дозволяє працювати із зображеннями, зберігаючи баланс між точністю та швидкодією.

Особливі точки (ключові точки) – це локальні ознаки зображення, які мають унікальні властивості та дозволяють ефективно співставляти об’єкти на різних зображеннях. Такі точки повинні виділятися серед сусідніх пікселів, бути стійкими до змін яскравості, контрасту, масштабу, повороту та шуму, а також бути представленими у зручному форматі для подальшої обробки [7, 23].

Виділення ключових точок виконується детектором, а дескриптор є описом околу кожної точки у вигляді чисел, що є параметрами, представленими у вигляді вектора. У такий спосіб можна однозначно ідентифікувати цю точку на інших зображеннях. При цьому в одному зображенні може бути виявлено багато ключових точок, кожна з яких має власний дескриптор [10].

Глобальні ознаки описують зображення цілком (наприклад, колір або текстуру), локальні ознаки натомість, формуються на основі ключових точок, а отже краще підходять для задач класифікації об’єктів, відстеження, порівняння сцен чи зображень одного й того ж об’єкта з різних ракурсів. Локальні дескриптори добре використовувати в умовах обмежених ресурсів або у реальному часі завдяки їх компактності та інформативності (рис. 1.4) [9, 24].

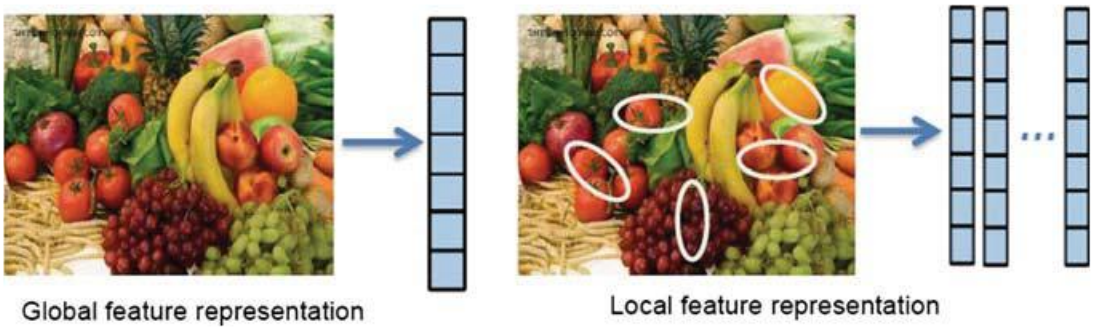


Рисунок 1.4 – Подання глобальних та локальних ознак зображення

У сучасному комп’ютерному зорі дескриптори умовно поділяють на два основні типи: дескриптори з плаваючою точкою та бінарні дескриптори [10].

Дескриптори з плаваючою точкою, такі як SIFT або SURF, зберігають інформацію про локальні області зображення у вигляді векторів з дійсними значеннями. Зазвичай вони засновані на розподілах градієнтів інтенсивності або напрямків контурів. Однак хоча такі дескриптори забезпечують високу точність і стійкість до змін умов зйомки, вони є обчислювально затратними, потребують значного обсягу пам’яті та не завжди підходять для пристроїв із обмеженими ресурсами, тобто мобільних пристроїв, наприклад [10].

На противагу їм, бінарні дескриптори, такі як BRIEF, ORB або BRISK, формують компактне представлення околиці ключової точки у вигляді бінарного рядка (рис. 1.5). Цей рядок отримується шляхом порівняння інтенсивностей пікселів у заздалегідь визначених точках шаблону навколо ключової точки. Такий підхід дозволяє значно знизити обчислювальну складність, забезпечує швидке зіставлення (завдяки використанню метрики

Геммінга) і зменшує споживання пам’яті, що робить бінарні дескриптори особливо привабливими для використання у реальному часі [10, 17, 25].

Проте через жорстко задану форму шаблону (наприклад, кругову чи прямокутну) бінарні дескриптори, зазвичай, не мають афінної інваріантності. Тобто, при складніших змінах ракурсу їх стійкість знижується, а це може підвищити ймовірність помилково знайдених відповідностей між об’єктами, які були зняті під різними кутами [10].

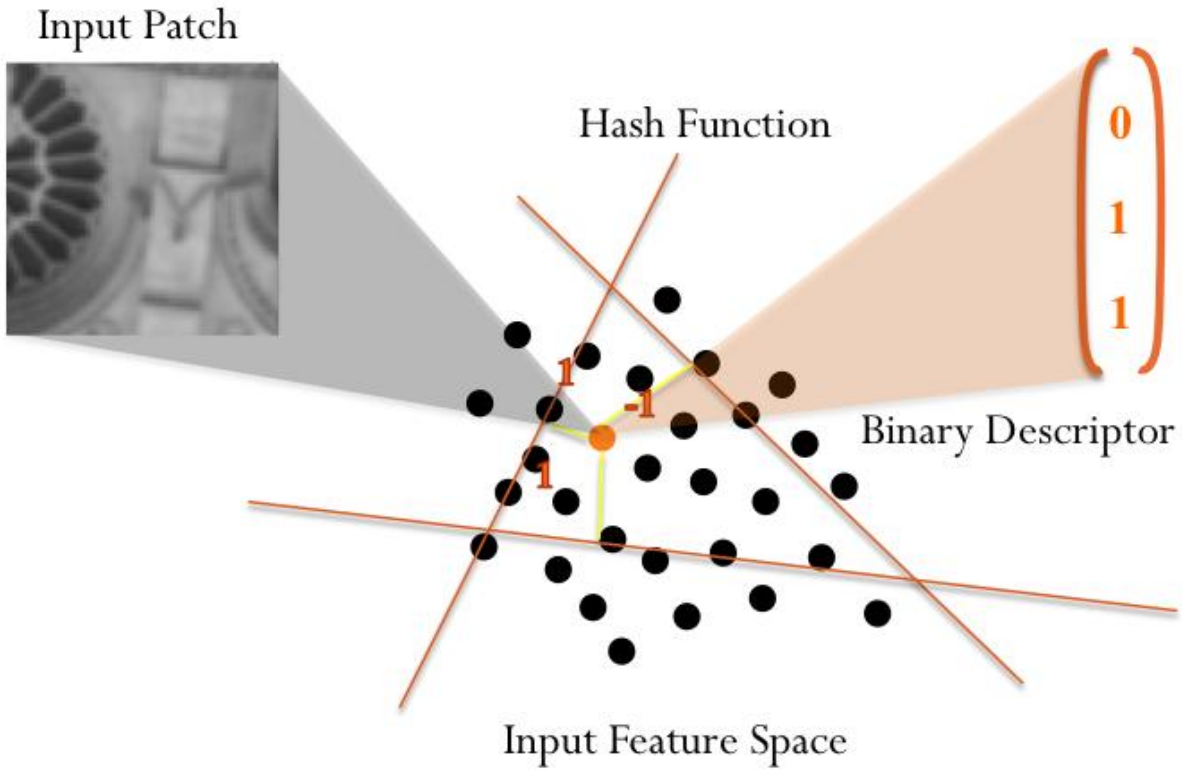


Рисунок 1.5 – Схема побудови бінарного дескриптора шляхом проєктування ознак у бітовий простір за допомогою порівняння інтенсивностей

Незважаючи на це, на практиці, при порівнянні або пошуку схожих зображень, класифікація чи відстежуванні об’єктів бінарні дескриптори демонструють високу результативність, особливо коли важлива швидкість і економне використання ресурсів [26, 27].

Оскільки в межах цієї роботи використовуються бінарні дескриптори, далі розглянемо метод BRISK (Binary Robust Invariant Scalable Keypoints), який вважається одним із найуспішніших методів цього типу [27].

BRISK – це детектор ключових точок, який дозволяє швидко обробляти зображення зі збереженням стійкості до змін масштабу. Його робота базується на вдосконаленому варіанті алгоритму AGAST (Adaptive and Generic Accelerated Segment Test), а для визначення ключових точок використовується оцінка FAST. Пошук особливих точок проводиться на зображенні представленому в різних масштабах, що дає змогу виявляти найкращі ознаки, які менш чутливі до змін розміру [28].

Для побудови дескриптора BRISK використовує заздалегідь визначений шаблон вибірки: точки розміщено по декількох концентричних колах навколо ключової (рис. 1.6). Такий підхід частково нагадує метод DAISY, але в BRISK він змінений відповідно до вимог швидкості та компактності, а не для щільного зіставлення. Щоб зменшити вплив шуму при зчитуванні інтенсивності, значення згладжуються Гаусовим фільтром, де ступінь згладжування залежить від відстані до центру [28, 29].

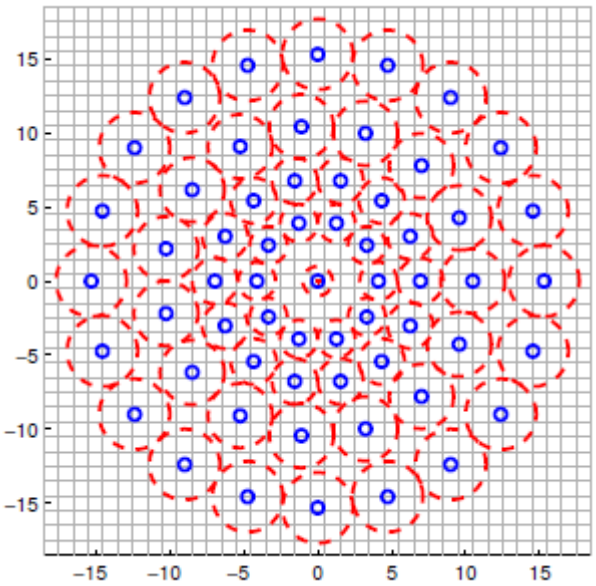


Рисунок 1.6 – Шаблон вибірки BRISK з розміщенням точок у кільцях навколо ключової точки

Пари точок у шаблоні умовно поділяються на два типи:

– довгі пари – використовуються для оцінювання орієнтації області навколо точки;

– короткі пари – застосовуються для побудови бітового дескриптора шляхом порівняння яскравості в точках пари [28].

Щоб отримати точний результат порівняння інтенсивностей, на кожній парі точок обчислюється локальний градієнт. Формула для обчислення градієнта між двома точками p_i і p_j виглядає так:

$$g(p_i, p_j) = (p_j - p_i) \cdot \frac{\left(I(p_j,\sigma_j) - I(p_i,\sigma_i)\right)}{\|p_j - p_i\|^2}, \quad (1.1)$$

де $g(p_i, p_j)$ – локальний градієнт між точками p_i і p_j;

$I(p, \sigma)$ – інтенсивність зображення в точці p, згладжена Гаусовим фільтром зі стандартним відхиленням σ;

$\|p_j - p_i\|$ – евклідова відстань між точками p_i і p_j [28].

Оскільки дескриптор має бути стійким до поворотів і змін масштабу, шаблон вибірки обертається на основі орієнтації ключової точки. Для цього визначають орієнтацію як арктангенс відносно компонент градієнта по осях x та y:

$$\alpha = arctan2(g_y, g_x), \quad (1.2)$$

де α – кут повороту шаблону;

g_y – компонента градієнта по осі y;

g_x – компонента градієнта по осі x [28].

Для створення бітового дескриптора порівнюються інтенсивності пікселів на парах точок. Якщо інтенсивність пікселя на точці p_j більша за інтенсивність пікселя на точці p_i, то встановлюється біт 1, в іншому

випадку – 0. Це дозволяє сформувати 512-бітовий дескриптор для кожної ключової точки. Формула для порівняння інтенсивностей точок і отримання бітового значення виглядає так:

$$b = \begin{cases} 1, \ I\left(p_j^\alpha, \sigma_j\right) > I\left(p_i^\alpha, \sigma_i\right) \\ 0, \ \text{в іншому випадку} \end{cases}, \ \forall\left(p_i^\alpha, p_j^\alpha\right) \in S, \tag{1.3}$$

де b – окремий біт у дескрипторі;

S – набір усіх пар точок для порівняння [28].

У результаті формується 512-бітовий дескриптор, який, як і BRIEF чи ORB, ґрунтується на бінарних порівняннях. Одночасно із цим BRISK має низку переваг: рівномірне розташування точок, обмежену кількість пар (що знижує обчислювальні витрати) та просторову узгодженість при вибірці. Завдяки таким характеристикам BRISK добре підходить для задач, де критичною є швидкість роботи, компактність ознак і стійкість до змін масштабу та повороту (рис. 1.7). Для порівняння дескрипторів, як і у BRIEF, застосовується відстань Геммінга [27, 29].

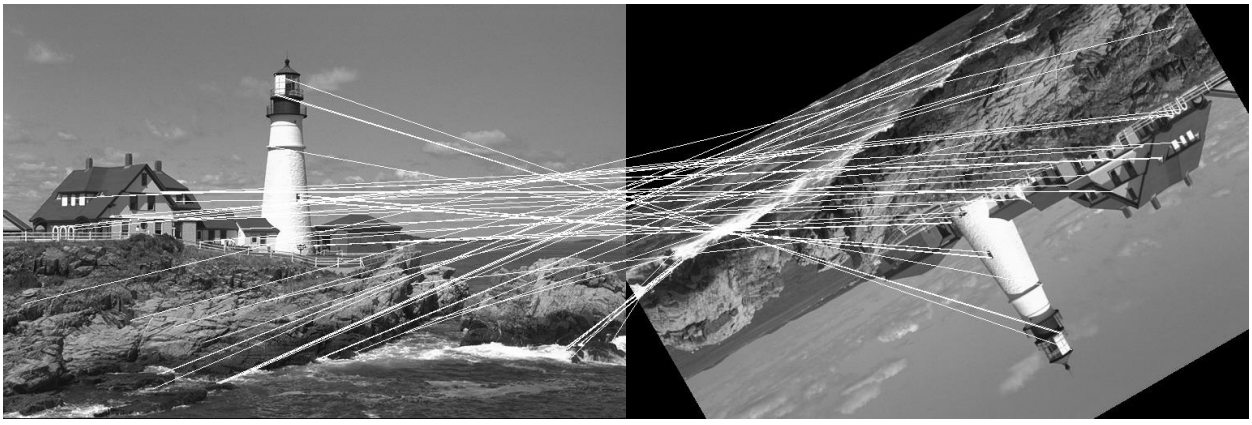


Рисунок 1.7 – Приклад зіставлення ключових точок методом BRISK між оригінальним і трансформованим зображенням

При реалізації BRISK бібліотека OpenCV надає розробникам гнучкість у налаштуванні роботи алгоритму відповідно до вимог конкретного завдання.

Зокрема, користувач може змінювати кілька основних параметрів при створенні об’єкта детектора:

– *thresh* – поріг, який використовується для виявлення кутів, і менше значення якого збільшує кількість виявлених точок, але також може призвести до більшої кількості слабких відповідей;

– *octaves* – кількість масштабних рівнів (октав), на яких виконується виявлення ключових точок, що забезпечує стійкість до змін масштабу;

– *patternScale* – масштабний коефіцієнт для шаблону вибірки, який визначає розміри області навколо ключової точки [30].

Кожна знайдена ключова точка в бібліотеці OpenCV представлена об’єктом *cv::KeyPoint*, який також має кілька характеристик. Зокрема, параметр *response* відображає силу виявленої точки, тобто ступінь її виразності серед навколишніх пікселів [31].

Такі параметри дозволяють адаптувати BRISK до різних застосувань, що робить його зручним інструментом для використання у різних задачах комп’ютерного зору.

1.3 Кластеризація та квантування ознак

Після виявлення ключових точок на зображенні наступним етапом є формування дескрипторів ознак, що описують локальні області поблизу знайдених точок. Ці дескриптори являють собою багатовимірні числові вектори, які кодують характерні властивості околу пікселів. Завдяки такому представленню, кожне зображення можна описати набором дескрипторів, що відповідають його локальним особливостям [7].

Проте кількість отриманих дескрипторів навіть з одного зображення може бути значною, а для великих наборів даних – надмірною. Це створює виклик щодо зберігання, порівняння та подальшої обробки. Відповідно, ключовим завданням стає перехід від множини локальних дескрипторів до

компактного векторного представлення зображення, яке б зберігало інформативність, але зменшувало обсяг даних для обробки. У зв’язку з цим ключовим питанням стає як найкраще організувати та подати такі дескриптори на подальшу обробку.

Одним із найпоширеніших підходів до класифікації зображень є модель «торба слів» (bag-of-visual-words model, модель BoVW), яка подає зображення як сукупність локальних ознак, розділених на кластери – «візуальні слова» (рис. 1.8). Цей метод запозичили із задач обробки текстів, де документи подають у вигляді гістограми входжень слів із фіксованого словника [23, 32-34].

Рисунок 1.8 – Витягування візуальних ознак і формування «торби слів»

Порівняно з індивідуальними дескрипторами, метод BoVW дозволяє значно зменшити кількість ознак, що підвищує результативність пошуку

найближчих сусідів і робить самі ознаки більш стійкими до змін. Алгоритм складається з таких основних етапів:

– виявлення ключових точок – на аналізованому зображенні знаходять локальні ключові точки;

– обчислення дескрипторів – навколо кожної точки формується числовий вектор, який описує властивості околу;

– побудова словника – усі дескриптори кластеризуються за допомогою, наприклад, алгоритму *k*-середніх. Центри кластерів розглядаються як «візуальні слова», а розмір словника визначається кількістю кластерів;

– формування гістограми – для кожного зображення підрахується кількість дескрипторів, які потрапили в кожен кластер;

– класифікація – отримані гістограми подають на вхід класифікатору, що забезпечує чітке розділення класів шляхом максимізації відстані до найближчих об’єктів кожної групи [32].

Процес кластеризації, що лежить в основі побудови візуального словника, полягає в об’єднанні подібних дескрипторів у групи – кластери. Кластеризація – це процес групування даних таким чином, щоб подібні об’єкти потрапляли в один кластер, тоді як відмінні – у різні. Основною метою є формування груп, в яких об’єкти мають високий рівень подібності між собою та суттєві відмінності від об’єктів інших груп. В якості метрики подібності найчастіше використовуються відстані у багатовимірному просторі ознак [32, 33].

Одним з найбільш популярних методів кластеризації є алгоритм *k*-середніх, який часто використовують у різних задачах обробки зображень. У контексті побудови візуального словника, *k*-середніх використовується для кластеризації дескрипторів зображень, що дозволяє перетворити їх у компактні представлення шляхом квантування – тобто заміни кожного дескриптора на центр відповідного кластера. Це не лише зменшує обсяг даних, а й підвищує стійкість до локальних варіацій та шумів, що, в свою чергу, сприяє покращенню точності класифікації [35, 36].

Основні етапи алгоритму *k*-середніх є такими:

Крок 1. Визначається кількість кластерів, на які буде розподілено дані.

Крок 2. Випадковим чином обираються k початкових центрів кластерів (центроїдів) зі всіх дескрипторів зображень.

Крок 3. Кожен дескриптор x призначається до найближчого центру кластера y на основі обраної метрики відстані, зазвичай евклідової:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}, \quad (1.4)$$

де x – окремий біт у дескрипторі;

y – набір усіх пар точок для порівняння;

n – набір усіх пар точок для порівняння.

Крок 4. Для кожного кластера C_i обчислюється новий центр як середнє значення всіх дескрипторів x_j, що були до нього призначені:

$$C_i = \frac{1}{|N_i|}\sum x_j, \quad (1.5)$$

де N_i – окремий біт у дескрипторі;

$|N_i|$ – набір усіх пар точок для порівняння.

Крок 5. Кроки 3 і 4 повторюються до тих пір, поки центри кластерів не перестануть змінюватися або не буде досягнуто максимальну кількість ітерацій [37, 38].

Цей ітеративний процес дозволяє ефективно групувати подібні дескриптори, що сприяє зменшенню розмірності даних та підвищенню стійкості до шумів і варіацій у зображеннях. Для якісного аналізу та перевірки результатів кластеризації часто використовують методи візуалізації, зокрема Т-розподілене вкладення стохастичної близькості (t-Distributed Stochastic Neighbor Embedding, t-SNE). Такий метод дає змогу зобразити високовимірні дані на площині з максимально можливим збереженням метричних

властивостей і полегшує інтерпретацію структури кластерів та виявлення закономірностей у розподілі дескрипторів (рис. 1.9) [39, 40].

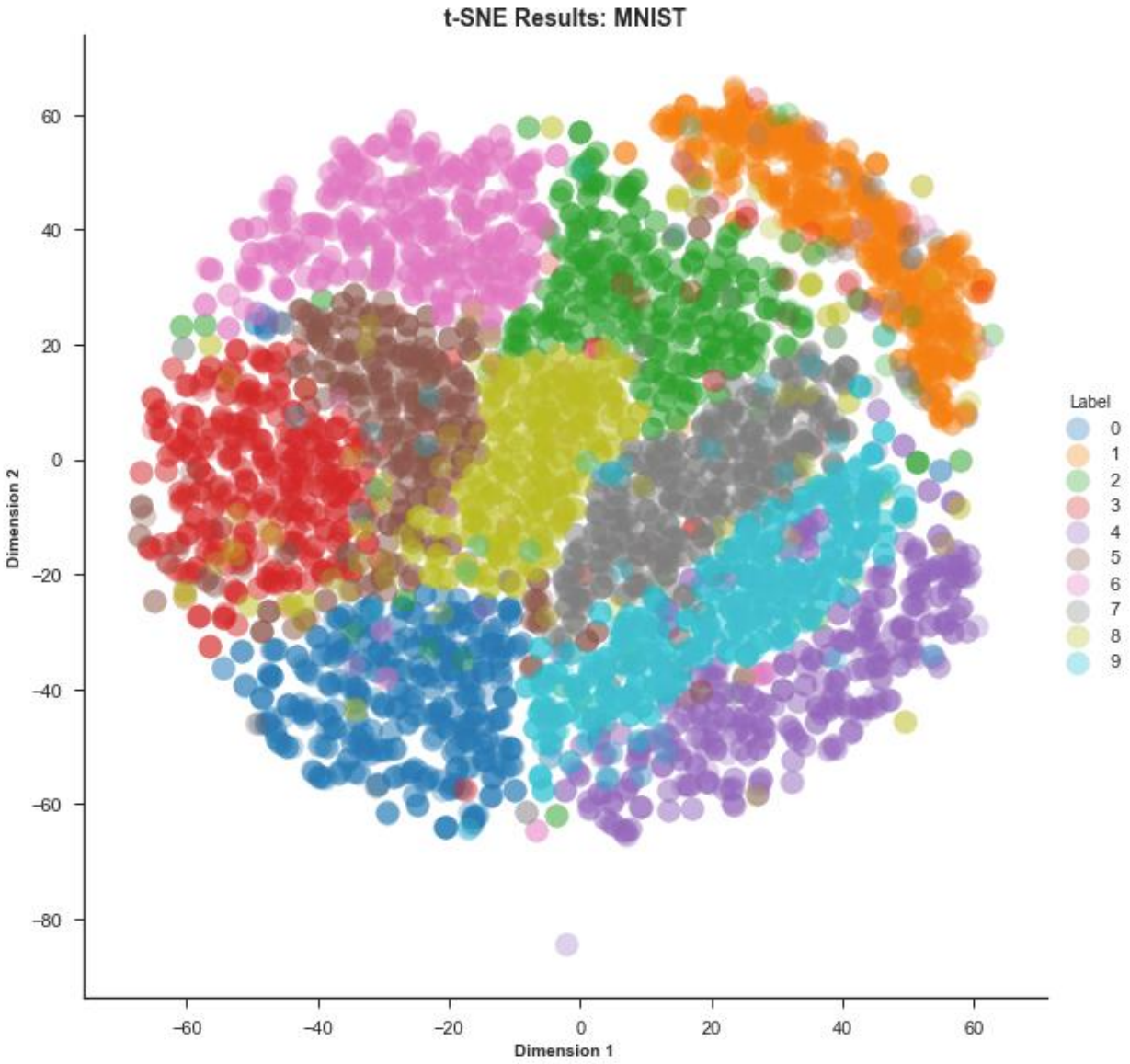


Рисунок 1.9 – Візуалізація кластерів у наборі даних MNIST методом t-SNE

Після кластеризації дескрипторів і формування центрів кластерів (візуальних слів), наступним етапом є квантування, яке дозволяє перетворити високовимірні ознаки у компактніші представлення. У загальному випадку квантування полягає в заміні значень дескрипторів на коди фіксованої довжини. Це значно знижує складність подальшої обробки, особливо в умовах великомасштабної класифікації, де актуальним стає питання економії обчислювальних ресурсів [11, 12, 21].

Цю проблему можна подолати шляхом проєктування початкового опису у формі множини векторів на новий, квантований простір даних. Такий простір може бути отриманий за допомогою кластеризації або інших методів квантування, зокрема шляхом застосування мережі Кохонена чи хешування [35]. У моделі, що базується на квантуванні до непересічних кластерів, кожен вектор автоматично ідентифікується з центром відповідного кластера. У результаті отримані підмножини дескрипторів не перетинаються, тобто формують розбиття множини описів. При цьому дані всередині одного кластера вважаються еквівалентними між собою та центру, що усуває неоднозначність у визначенні подібності між векторами.

Такий підхід фактично виконує попередню класифікацію у квантованому просторі, дозволяючи надалі застосовувати формалізований апарат метричних оцінок для аналізу релевантності. Пропоновану зміну структури опису зображень можна трактувати як формування трансформованого простору ознак, у якому забезпечується простіше та ефективніше розділення образів. Це відповідає фундаментальним положенням теорії розпізнавання образів, згідно з якими успішне вирішення прикладних задач можливе за умов зниження розмірності простору та спрощення вирішальних правил [36].

У новоутвореному квантованому просторі багатовимірних даних для оцінювання перетину двох описів у вигляді множин зручно використовувати апарат нечітких множин або мультимножин. У цьому випадку центри кластерів виступають базовими елементами мультимножинного представлення або значеннями нечіткої множини. Еталонні описи можна попередньо подати у вигляді мультимножин – цілочисельних векторів, що містять кількість дескрипторів у кожному з кластерів. Процес класифікації тоді зводиться до зіставлення опису вхідного об’єкта з такими проєкціями еталонів або безпосередньо з центрами кластерів [15].

Додатковою перевагою цього підходу є значне прискорення класифікації, оскільки вдається уникнути обчислювально затратних процедур

лінійного пошуку за моделлю найближчого сусіда [17]. Крім того, запровадження квантування дозволяє уникнути трудомісткого голосування між дескрипторами різних множин, яке хоч і є поширеним способом оцінки подібності, але не зберігає властивості симетричності. Застосування метрик у квантованому просторі відкриває також можливості для використання розмаїття моделей метричних просторів, таких як коефіцієнт кореляції та інші.

Серед подібних підходів особливу увагу заслуговує метрика Танімото, яка є однією з найуживаніших для оцінювання подібності між множинами. На відміну від ряду інших метрик, вона забезпечує інтегральну оцінку релевантності двох множин A, B з урахуванням усієї сукупності їх компонентів [37]. До того ж, метрика Танімото допускає врахування повторень елементів, тобто застосовується і для мультимножин.

Традиційне визначення цієї метрики має вигляд:

$$\mu(A, B) = card(A \Delta B)/card(A \cup B). \quad (1.6)$$

У той же час (1.6) можна подати у більш практичній формі

$$\mu(A, B) = \frac{card(A)+card(B)-2card(A \cap B)}{card(A)+card(B)-card(A \cap B)}. \quad (1.7)$$

Як видно з (1.7), основним етапом при обчисленні цієї метрики є визначення кількості спільних елементів у двох скінченних множинах, тобто оцінювання потужності для їх перетину. Індивідуальні ж потужності зазвичай відомі заздалегідь, оскільки відповідають кількості дескрипторів у відповідних описах.

Метрика дозволяє ефективно реалізувати обчислення навіть у великих системах, що містять тисячі порівнянь, не втрачаючи продуктивності.

1.4 Постановка задачі

Розроблення методу структурної класифікації зображень з використанням квантування у просторі ознак є актуальним завданням комп’ютерного зору, зокрема для обробки зображень у реальному часі.

Об’єктом роботи є методи класифікації зображень за множиною дескрипторів ключових точок.

Метою роботи є розроблення модифікації для методу структурної класифікації зображень на основі квантування у просторі ознак, що передбачає застосування бінарних дескрипторів BRISK, методу *k*-середніх та метрики Танімото для швидкісного оцінювання релевантності об’єктів.

Для досягнення мети необхідно вирішити такі завдання:

– аналіз традиційних методів класифікації зображень з використанням ключових точок та дескрипторів;

– реалізувати алгоритм кластеризації опису зображень із отриманням центрів даних;

– реалізувати побудову даних опису у формі векторів ознак;

– здійснити моделювання традиційного методу класифікації – методу голосування;

– здійснити моделювання методу класифікації на основі метрики Танімото;

– провести тестування реалізованих методів та оцінити доцільність використання розроблених методів на основі параметрів точності та швидкодії.

2 КЛАСИФІКАЦІЯ ЗОБРАЖЕНЬ ТА ВИБІР ПАРАМЕТРІВ

2.1 Формальна постановка задачі класифікації

В задачах класифікації зображень на основі локальних ознак одним з інтуїтивно простих і водночас достатньо точних підходів є метод голосування, що ґрунтується на зіставленні дескрипторів між аналізованим зображенням і еталонами. Такий підхід особливо добре працює з бінарними дескрипторами (наприклад, BRIEF, ORB, BRISK), оскільки дозволяє швидко порівнювати вектори за допомогою відстані Геммінга та накопичувати статистику збігів.

Ідея методу полягає у тому, що кожен дескриптор вхідного зображення «голосує» за той еталон, дескриптор якого виявився найближчим за обраною метрикою (для бінарних дескрипторів – відстань Геммінга). Якщо відстань до найближчого дескриптора в еталоні менша за поріг, голос вважається «зарахованим». За підсумками таких голосувань підраховується кількість збігів з кожним класом, і на основі цього приймається рішення про належність об’єкта до певної категорії [11].

Будемо здійснювати класифікацію в рамках бази із N еталонів (представники чи прототипи класів), що задана у формі деякої скінченної множини E описів еталонних зображень: $E = \{E_1, E_2, \dots, E_N\}$. Множина E – це навчальна вибірка, яка одночасно є підґрунтям для побудови класифікатора, в тому числі шляхом зіставлення описів аналізованих об’єктів та еталонів [12].

Кожен еталонний опис E_k у формалізмі класифікатора репрезентує окремий клас. Клас k для з еталонним описом E_k формально розуміємо як деяку нескінченну множину зображень, отриманих із еталонного зображення (прототип класу з номером k) шляхом застосування до еталону багатопараметричної групи геометричних перетворень, яка найчастіше у прикладних застосуваннях включає зміщення, поворот, масштабування, дія яких не виводять об’єкт інтересу із поля зору [21].

Тепер визначимо простір B^n усіх бінарних векторів розмірності n. Відомо, що $card\, B^n = 2^n$. Будемо ототожнювати B^n із простором дескрипторів ключових точок (КТ) зображення, отриманих деяким детектором КТ [13]. Опис аналізованого об'єкту Z як і опис окремого еталону $E_k = \{e_v(k)\}_{v=1}^{s}$, $E_k \subseteq B^n$ будемо розглядати як скінченну множину потужністю s у просторі B^n бінарних векторів, $e_v(k) \in B^n$, $s = card E_k$ – число дескрипторів у множині. Усякий дескриптор $e_v(k)$ у складі бази E характеризується параметром k номеру класу. Загалом число ознак – дескрипторів у базовій множині E складає $card\, E = sN$. Число s дескрипторів для спрощення вважаємо однаковим для усіх E_k.

Процес R структурної класифікації для аналізованого об'єкту $Z = \{z_v\}_{v=1}^{s}$, $z_v \in B^n$, традиційним методом на підставі процедури голосування здійснимо у два етапи як $R = R_2 R_1$. Етап R_1 реалізує локальне рішення стосовно визначення класу для окремого дескриптора z_v об'єкту Z, а етап R_2 на підставі рішень (голосів) компонентів усього складу опису Z визначає значення параметра класу для аналізованого об'єкту. Фактично етап R_1 у такій моделі реалізує метод найближчого сусіда на навчальній множині E, де номер класу для дескриптора об'єкту визначається як клас найближчого елементу із E. А етап R_2 реалізує процедуру оцінювання релевантності описів за моделлю однорідного голосування із рішенням за найбільшим числом голосів на наборі класів.

Класифікатор R_1 за методом лінійного пошуку здійснює по-елементний аналіз вхідного опису $Z = \{z_v\}_{v=1}^{s}$ об'єкта і відносить кожний дескриптор $z_v \in Z$ до одного із класів за правилом

$$R_1 : z_v \to 1, \dots, N. \tag{2.1}$$

Клас k аналізованого дескриптора z_v у моделі (2.1) визначимо як аргумент мінімуму відстані на множині E

$$R_1 : k = arg \min_{i=1,\dots,N;\ d=1,\dots,s} \rho\left(z_v, e_d(i)\right). \quad (2.2)$$

Тут $\rho: B^n \times B^n \to [0, n]$ – відстань у векторному просторі B^n. Найбільш дієвим у просторі B^n бінарних векторів виглядає застосування у (2.2) метрики Геммінга [21].

Фактично модель класифікації (2.2) не тільки визначає клас k для дескриптора z_v об'єкту, а також встановлює відповідність для пари елементів $z_v \to e_d(k)$ множин Z та E_k. Цю відповідність можна описати як бінарне відношення еквівалентності між елементами $z_v, e_d(k)$ двох множин дескрипторів. При цьому допускається, що деякі елементи Z при реалізації (2.2) можуть взагалі не бути віднесені до жодного із еталонних класів, якщо отримана відповідність незначуща. Значущість відповідності для дескрипторів встановлюється перевіркою логічного правила [22]

$$\rho_m = \min_{i=1,\dots,N;\ d=1,\dots,s} \rho\left(z_v, e_d(i)\right),\ \rho_m \le \delta_\rho, \quad (2.3)$$

де δ_ρ – деякий поріг для мінімуму ρ_m метрики ρ.

Умова (2.3) встановлює еквівалентність для пари дескрипторів із врахуванням допустимого ліміту δ_ρ. Фактично за правилом (2.3) еквівалентність будь-якої пари дескрипторів визначається з деяким допуском, а кожен дескриптор z_v у просторі B^n розглядається як деяка багатовимірна куля, яка містить еквівалентні до z_v дескриптори z_x за моделлю $\rho(z_v, z_x) \le \delta_\rho$.

Здійсненням R_1 для кожного $z_v \in Z$ відповідно до (2.2), (2.3) визначимо номер класу k, а потім інкрементуємо вектор – акумулятор голосів $h_k = h_k + 1$ для отриманого номеру класу за моделлю (2.2). Фактично обчислений агрегацією за множиною елементів опису об'єкта вектор $\{h_i\}_{i=1}^N$ є розподілом (гістограмою) числа голосів дескрипторів об'єкту у наявній системі класів. Приклад розподілу числа голосів для зображень наведений на рисунку 2.1.

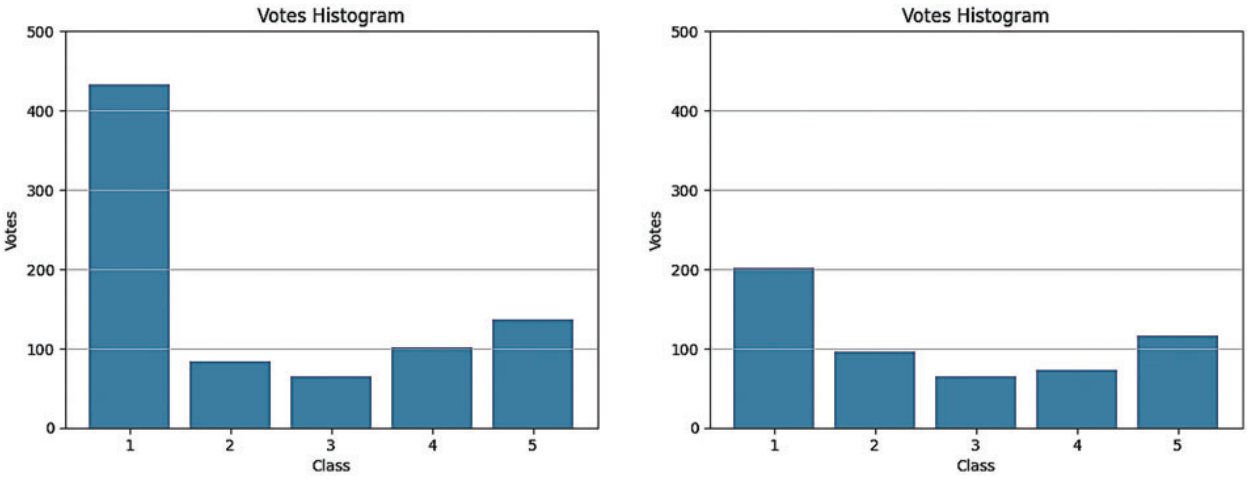


Рисунок 2.1 – Приклади гістограм числа голосів дескрипторів об’єктів

Побудуємо тепер правило R_2 на підставі вектору $\{h_i\}_{i=1}^{N}$ – акумулятора оцінок класів з цілими значеннями для накопиченої кількості голосів, отриманих застосуванням моделі (2.2) до усієї множини дескрипторів об’єкту Z. Клас об’єкту визначимо правилом R_2 як аргумент максимуму

$$R_2: \mathrm{Z} \to E_k | (k = arg \max_{i=1,\dots,N} h_i) \,\&\, (h_k \geq \delta_h), \tag{2.4}$$

де δ_h – деякий поріг для мінімально допустимого значення максимуму серед числа голосів.

Якщо умова $h_k \geq \delta_h$ не виконується, клас об’єкту не встановлюється (відмова від класифікації, немає підстави віднести об’єкт до жодного із наявних класів).

Значення δ_h визначають експериментально для заданої бази E еталонів. Як правило, його обирають як мінімальне число голосів (з допуском), необхідне для впевненої класифікації тестової вибірки на основі трансформацій еталонів. Із загальних теоретичних позицій δ_h є компромісом у задачі розрізнення скінченного набору еталонів від практично неосяжної

решти інших зображень, що можуть поступити на вхід системи класифікації [10].

Послідовність правил класифікації R_1, R_2 реалізує класифікатор на підґрунті ансамблю рішень набору класифікаторів, отриманих для множини складових компонентів об’єкту. Він впроваджує принципи структурного аналізу і узгодження із базою еталонів, а також забезпечує стійкість до просторових викривлень окремих компонентів (дескрипторів) із-за можливого впливу завад і фону у процесі аналізу зображень. Схему класифікації методом голосування наведено на рисунку 2.2.

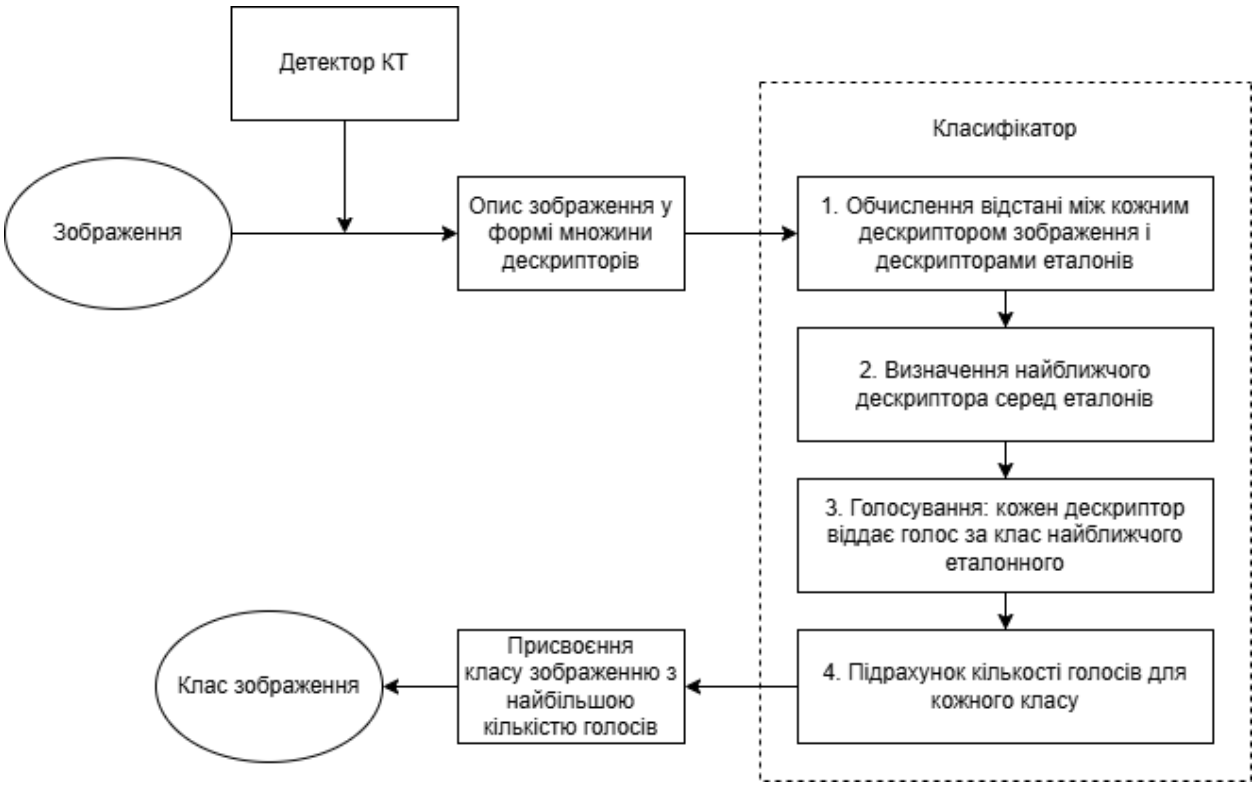


Рисунок 2.2 – Схема класифікації зображень методом голосування

Результативність класифікації будемо традиційно оцінювати критерієм точності pr, який відображає відношення кількості q тестових описів з правильною оцінкою класу до загального числа Q проведених експериментів

$$pr = q/Q. \tag{2.5}$$

Таким чином, метод голосування на основі попарного порівняння дескрипторів демонструє високу точність класифікації зображень. Його стійкість до просторових викривлень, здатність працювати з бінарними ознаками та прозора логіка прийняття рішення роблять його привабливим для задач, де важлива інтерпретованість результату та надійність при наявності завад.

Водночас слід зазначити істотний недолік моделі – її низьку швидкодію. Через необхідність виконання великої кількості попарних порівнянь між дескрипторами аналізованого зображення та усіх еталонів, час обробки одного зображення може бути значним, особливо при великому розмірі бази. Це обмежує застосування методу у задачах, що потребують реального часу або обробки великих потоків даних.

Попри це, модель залишається ефективною для офлайн-аналізу, прототипування або як еталон для порівняння з іншими методами класифікації в рамках експериментального дослідження. Подальші напрями оптимізації можуть включати скорочення кількості порівнянь шляхом попередньої фільтрації, застосування структурованих пошуків (наприклад, з використанням дерева пошуку) або використання апроксимаційних методів для пришвидшення обчислень.

2.2 Класифікація з використанням метрики Танімото

У контексті задач класифікації візуальних об’єктів важливим є не лише точне, а й компактне подання інформації про зображення. Одним із ефективних підходів є використання кластеризації для представлення великої кількості дескрипторів у стислому вигляді, що дозволяє зменшити обчислювальні витрати. Такий підхід, відомий як квантування ознак, полягає у перетворенні множини дескрипторів на вектори частотного розподілу, які

узагальнюють інформацію про структуру зображення через систему кластерів (центрів квантів).

Метод базується на ідеї перетворення простору дескрипторів у дискретизовану форму, де кожен дескриптор відноситься до певного кластеру за конкурентним правилом. Це дозволяє використовувати моделі типу «торба слів», в яких образи описуються за частотою появи типових ознак. Далі ці вектори подаються на вхід класифікатору, який приймає рішення на основі метричних співвідношень між об'єктами та еталонами у новому просторі.

Здійснимо квантування (кластеризацію) для спільного опису повної бази E із N еталонів E_k на M квантів. Квантування реалізується процедурою самонавчання і призначено для подання наявної множини еталонних даних у формі фіксованого набору кластерів, що містять подібні (еквівалентні, близькі, рівноцінні) між собою елементи. Це своєрідна сегментація (розбиття) множини дескрипторів наявної бази еталонів на M груп.

У результаті отримаємо систему центрів $W = \{w_j\}_{j=1}^{M}$ квантів і подання бази E у формі розбиття $E = \cup_{j=1}^{M} T_j$, де складові сегменти даних – кластери T_j отримані так, що не перетинаються між собою. При цьому центри w_j в залежності від процедури самонавчання можуть взагалі не належати простору B^n, але їх можна трансформувати в B^n, наприклад, шляхом округлення компонентів w_j. З точки зору теорії мультимножин отримані центри створюють деяку базову множину, на яку тепер можна універсально здійснити проектування довільної множини бінарних векторів із простору B^n.

Спочатку здійснимо проектування за схемою [20] для складу наявних еталонних описів. У відповідності до схеми проектування для довільного дескриптора $z \in E_k$ еталону чи об'єкта встановлюється номер j кластера (кванта) за конкурентним правилом

$$z \in T_j \mid j = arg \min_{i=1,\dots,M} \rho(w_i, z). \qquad (2.6)$$

Правило (2.6) реалізує один із варіантів моделі класифікації «торба слів», де базовою множиною виступає множина центрів кластерів [20]. У виразі (2.6) при $w_j \in B^n$ можна використати метрику Геммінга. Схематичне представлення побудови векторів ознак на основі кластеризації ключових точок наведено на рисунку 2.3 [23].

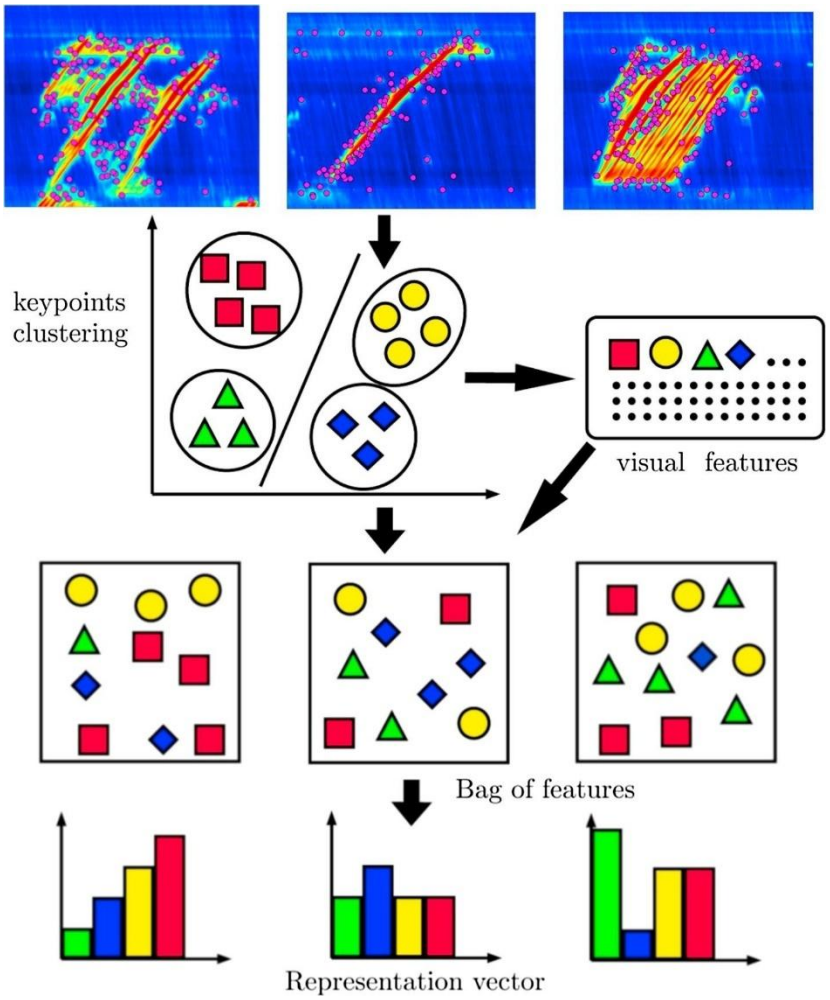


Рисунок 2.3 – Схема побудови векторного подання зображень на основі кластеризації ключових точок у моделі «торба візуальних слів»

У результаті застосування (2.6) до усієї множини компонентів опису чисельно отримуємо подання кожного опису E_k у формі вектора цілих чисел $\varphi[E_k] = \{\varphi_{k,i}\}_{i=1}^{M}$ з числом компонентів, що відповідає числу M кластерів. Значення $\varphi_{k,i}$ – це кількість елементів еталону E_k, віднесених в результаті аналізу (8) до кластера з номером j. Отримані дані кластерного подання для

бази E можна розглядати як матрицю $\{\{\varphi_{k,i}\}_{i=1}^{M}\}_{k=1}^{N}$, в рядках якої міститься кількісне розкладення складу еталону за системою кластерів, а у стовпцях – чисельне представлення складу кожного кластера за системою класів.

Тепер здійснимо проектування шляхом застосування правила (2.6) до множини елементів $z_v \in Z$ аналізованого об'єкту. Отримаємо чисельне подання як вектор $\varphi[Z] = \{\varphi_i(Z)\}_{i=1}^{M}$. Цей вектор можна трактувати як функцію належності елементів об'єкту до множини кластерних центрів. Схему отримання подання опису зображення у формі вектора представлено на рисунку 2.4.

Рисунок 2.4 – Трансформація опису зображення за результатом квантування

Використаємо моделі метрик для обчислення потужності (2.7), перетину (2.8), об'єднання (2.9) та симетричної різниці (2.10) для пари мультимножин A, B (еталон – об'єкт) [21]

$$card(Z) = \sum_{j=1}^{M} \varphi_j, \tag{2.7}$$

$$card(A \cap B) = \sum_{j=1}^{M} min\,[\,\varphi_j(A), \varphi_j(B)], \tag{2.8}$$

$$card(A \cup B) = \sum_{j=1}^{M} max\,[\,\varphi_j(A), \varphi_j(B)], \tag{2.9}$$

$$card(A \Delta B) = \sum_{j=1}^{M} |\,\varphi_j(A) - \varphi_j(B)|\,. \tag{2.10}$$

Ці метрики є теоретико-множинним еквівалентом для перетину та об'єднання двох множин. Далі визначимо величину метрики $\mu\,(A,B)$ Танімото за виразами (2.6) чи (2.7). Параметр M тут виступає як розмірність порівнюваних мультимножин [15]. Співвідношення (2.7) – (2.10) використовують у теоріях нечітких множин і трикутних норм [21].

Також для отримання функції належності до множини – перетину двох множин з нормованими функціями належності використовують такі моделі, як алгебраїчний добуток $\varphi_j(A) \times \varphi_j(B)$ (по-елементне перемноження значень функцій належності) та обмежену суму $max\,[\,\varphi_j(A) + \varphi_j(B) - 1,\ 0]$. Усі ці дії над множинами більш ґрунтовно розглядають у застосуваннях теорії трикутних норм. Трикутна норма – це двомісна функція $f\,(x,y), f\colon [0,1]^2 \to [0,1]$ із областю визначення у вигляді замкнутого одиничного квадрату і множиною значень у формі замкнутого одиничного інтервалу [21].

Результуюче класифікаційне рішення про клас об'єкту з описом Z приймаємо як аргумент мінімуму відстані на множині $\{\varphi[E_i]\}$ для проекцій еталонів

$$R_2\colon Z \to E_k \mid k = arg \min_{i=1,\dots,N} \mu\,(\varphi[Z], \varphi[E_i])\,. \qquad (2.11)$$

Традиційний метод класифікації з використанням голосування реалізує послідовність моделей (2.2) – (2.4).

Класифікатор з використанням квантування на основі системи центрів W спочатку отримує за моделлю (2.6) векторне подання $\varphi(Z) = [\varphi_1, \varphi_2, \dots, \varphi_M]$ для розпізнаваного об'єкту, а далі з використанням метрики μ для обчислення релевантності векторів цілих чисел визначає клас об'єкту як аргумент мінімуму метрики μ на множині кластерних подань для еталонів бази E.

Варіантом метрики μ в (2.11) може бути мангеттенська чи будь-яка інша відстань для простора векторів цілих чисел. Використання правила (2.11) в методах з кластеризацією замість (2.2), (2.4) для традиційного методу дає виграш у обсязі обчислень приблизно пропорційний величині $\gamma = s/M$. Відмітимо, що задля забезпечення відсіювання сторонніх зображень обчислене оптимальне значення метрики μ у (2.11) повинне задовольняти умові значущості $\mu_v \leq \delta_\mu$.

При практичному застосуванні метричного апарату для класифікації треба звернути особливу увагу на визначення центру кластера за правилом (2.6) при формуванні $\varphi[Z]$. Отриманий у (2.6) мінімум метрики ρ_j обов'язково повинен задовольняти умові (2.3) значущості $\rho_j \leq \delta_\rho$. Ця вимога необхідна на практиці для відсіювання хибних дескрипторів, що появляються під впливом фону [21]. Загалом пороги для значень метрики і числа голосів визначаються експериментально, вони залежать від бази зображень і умов розпізнавання.

Ще одним практичним обмеженням є вимога щодо приблизно однакового числа дескрипторів у описах об'єкту та еталонів. Якщо потужності множин значно різняться, необхідно додатково нормувати дані за числом дескрипторів для здійснення виваженого прийняття класифікаційного рішення. У той же час використання метрики Танімото дає можливість здійснити класифікацію без додаткового нормування даних.

У квантованому векторному просторі можна застосувати і інші міри подібності, наприклад, коефіцієнт кореляції.

Застосування кластеризації для подання дескрипторів дозволяє значно зменшити розмірність даних, а також забезпечує швидшу обробку в порівнянні з класичним попарним порівнянням дескрипторів. Завдяки цьому суттєво скорочується час класифікації, що особливо важливо при роботі з великими базами зображень. Кластерне подання є компактним, універсальним і дозволяє застосовувати широкий спектр метричних функцій, включаючи метрику Танімото, мангеттенську відстань, а також функції, засновані на теорії нечітких множин.

Однак для забезпечення коректності класифікації необхідно дотримуватися ряду умов – зокрема, контролювати якість центрування кластерів, нормування описів за кількістю дескрипторів, а також використовувати порогові значення метрик для відсіювання фонового шуму та нерелевантних об’єктів. Успішна реалізація цього підходу вимагає ретельної параметризації на основі експериментальних даних для кожної конкретної задачі.

Таким чином, класифікація у векторизованому кластерному просторі поєднує в собі переваги високої узагальненості, гнучкості метрик та знижених обчислювальних витрат, що робить її ефективним інструментом для практичного розпізнавання зображень.

2.3 Обґрунтування щодо визначення параметрів класифікації

При використанні моделі класифікації з квантуванням даних особливого значення набуває механізм зіставлення дескрипторів з центрами кластерів та подальше формування векторного представлення образу. Оскільки класифікація базується на метричних відстанях між такими поданнями, важливою є оптимізація параметрів, що контролюють відбір дескрипторів і точність представлення зображень у просторі ознак.

Зауважимо, що обговорюваний процес класифікації з використанням метричних співвідношень між кластерними поданнями спирається на ряд параметрів, найбільш важливими серед яких є пороги:

– δ_ρ – для встановлення еквівалентності пари дескрипторів;

– δ_h – для вирішального значення максимуму числа голосів класів;

– δ_μ – для мінімуму значення метрики між кластерними описами.

Пороги δ_h, δ_μ визначають необхідний накопичений ступінь значущості при прийнятті рішення про клас об’єкту і можуть бути оцінені за результатом аналізу апріорної інформації у базі класифікації. У більшості застосувань поріг

δ_ρ традиційно обирають як 25% (чи менше) від максимуму значення метрики ρ для пари дескрипторів [28].

Із врахуванням специфіки методу класифікації на підставі використання кластерного подання введемо новий поріг $\delta_{\rho 1}$ для відстані між дескриптором і центром кластеру, який впровадимо у процедурі (2.6) при визначенні оптимального за відстанню центру кластера для дескрипторів аналізованого опису. Якщо мінімум відстані у моделі (2.6) не перевищує значення $\delta_{\rho 1}$, то не будемо відносити дескриптор до жодного із центрів кластерної системи. Цю умову можна формалізувати шляхом перевірки істинності деякого предиката $Pr(z, \delta_{\rho 1})$ двох аргументів:

$$Pr(z, \delta_{\rho 1}) : \min_{i=1,\dots,M} \rho(w_i, z) \leq \delta_{\rho 1}. \qquad (2.12)$$

Цей предикат приймає значення одиниці при виконанні нерівності в (2.12) і нулю – у протилежному випадку. Таким чином ми фактично вводимо процедуру фільтрації хибних дескрипторів за критерієм віддаленості від системи центрів, на яку спирається класифікація.

У запропонованому варіанті оброблення (2.12) поріг $\delta_{\rho 1}$ буде мати досить серйозний вплив на результат класифікації, так як його впровадження дасть можливість певною мірою відсіяти хибні дескриптори, які знаходяться віддалено від встановленої системи центрів. Ступінь такого відсіювання регулюємо параметром $\delta_{\rho 1}$. Значення параметру $\delta_{\rho 1}$ при цьому може змінюватися в межах діапазону значень метрики ρ.

Зауважимо, що нововведення (2.12) можна попередньо також застосувати і до еталонних описів. У результаті фільтрації кластерні представлення еталонів можуть змінитися в сторону скорочення, і, можливо, їх потужність стане неоднаковою для різних еталонів. Ця ситуація автоматично враховується використанням метрики Танімото. Інший шлях – це впровадження нормування для модифікованого еталонного подання на

загальну кількість дескрипторів опису, що задовольняють виставленій умові з порогом $\delta_{\rho 1}$.

Ще одним способом фільтрації даних, яку можна застосувати також і до складу еталонних описів, може бути використання моделі «чемпіонського списку» для даних всередині кожного із створених кластерів. Для цього визначимо деяку процедуру $D(T_j, w_j)$ оброблення даних, що відбирає фіксоване число d елементів кожного кластеру T_j із найближчою відстанню до його центру w_j. У результаті число елементів у квантованих еталонних описах скоротиться, буде виконано умову $card(T_j) = d$, а загальний обсяг еталонних даних буде дорівнювати Md як добуток числа M кластерів на зафіксоване число d елементів у кластері. Така модифікація скорочує обсяг кластерного подання.

Змінюючи процедурою D число d даних у кластері чи варіюючи порогом $\delta_{\rho 1}$, ми маємо можливість керувати обсягом даних у кластерному поданні. При стисненому обсязі даних можна оновити центри кластерів, щоб вони точніше апроксимували розподіл за класами, показники якого використовують у процесі класифікації [29].

Фактично таким нововведенням ми продукуємо D як процедуру для фільтрації простору дескрипторів за принципом їх узгодженості із системою кластерних центрів. Процедура D призводить до стиснення кластерного подання і деякого перерозподілу чисельності елементів різних класів всередині кластеру. Це дає можливість класифікатору краще адаптуватися до наявних даних задля покращення ефективності. Зауважимо, що процедуру D можна реалізувати на етапі підготовки, її реалізація безпосередньо не впливає на час класифікації.

Зрозуміло, що можна розглядати різні схеми впровадження пропонованого нововведення (2.12), в тому числі і спосіб його застосування виключно для фільтрації аналізованих даних вхідного опису, залишаючи незмінним результат початкової кластеризації бази еталонів.

Запропонований підхід до фільтрації дескрипторів дозволяє не лише зменшити кількість шумових та слабко інформативних ознак, а й підвищити якість класифікації за рахунок покращення узгодженості між вхідним описом і еталонною базою.

Загалом, застосування процедур фільтрації та нормування забезпечує адаптивність моделі до різнорідних даних, знижує ризик переобчислення та зменшує ресурсні витрати на класифікацію. Це особливо важливо в умовах обмежених обчислювальних можливостей або при роботі з великими обсягами візуальної інформації. Таким чином, запропоновані модифікації розширюють можливості моделі кластерного подання та підвищують її ефективність у практичних застосуваннях.

3 РЕЗУЛЬТАТИ КОМП'ЮТЕРНОГО МОДЕЛЮВАННЯ МЕТОДІВ КЛАСИФІКАЦІЇ

3.1 Обґрунтування програмного середовища

У рамках кваліфікаційної роботи було реалізовано програму для класифікації зображень із використанням двох методів: методу голосування на основі дескрипторів та методу з використанням метрики Танімото. Програмна реалізація виконана мовою програмування Python, яка є однією з найбільш часто використовуваною у галузі обробки зображень, аналізу даних та розробки систем штучного інтелекту. Популярність Python зумовлена його простим синтаксисом і широким вибором бібліотек, що робить його ідеальним вибором для таких задач [38, 39].

Python має величезну кількість бібліотек, що відкривають широкий спектр можливостей перед розробниками: від базової роботи з масивами до розв'язання складних задач машинного навчання та комп'ютерного зору. У проєкті було використано такі основні бібліотеки:

– OpenCV (cv2) – бібліотека комп'ютерного зору, яка містить сотні алгоритмів для виявлення, розпізнавання, обробки зображень і відео, а також підтримує детектори ключових точок, такі як BRISK [40];

– NumPy та Pandas – дозволяють працювати з числовими масивами та табличними даними;

– Matplotlib – використовується для побудови графіків, що дозволяє візуалізувати результати класифікації;

– Scikit-learn (sklearn.manifold) – застосовується для візуалізації кластерної структури за допомогою алгоритму зниження розмірності t-SNE;

– Openpyxl – дозволяє зберігати результати класифікації у форматі Excel;

– subprocess, threading, os, time – стандартні модулі для запуску процесів, керування потоками, файлової структури та синхронізації.

Для побудови графічного інтерфейсу користувача було використано бібліотеку CustomTkinter. Це обгортка над стандартною бібліотекою Tkinter, яка надає розробнику більше гнучкості в реалізації дизайнерських ходів, а також більший функціонал. CustomTkinter надає набір просунутих віджетів, таких як кнопки, поля введення та смуг прокручування, які можна налаштовувати за допомогою параметрів, як-от текст, колір фону, колір межі, колір при наведенні, шрифт та інші. Ці елементи забезпечують сучасний вигляд застосунку за задоволення при взаємодії з ним [41, 42].

Програму було розроблено у середовищі Visual Studio Code – сучасному кросплатформному редакторі коду з підтримкою Python.

Розробка та тестування програми здійснювалися в операційній системі Windows 11, що забезпечує сумісність з більшістю користувацьких систем та дозволяє використовувати специфічні функції, такі як відкриття папок через провідник Windows.

Крім того, реалізований інтерфейс дозволяє користувачеві зручно запускати процес класифікації, переглядати результати та зберігати їх у відповідному форматі. Такий підхід значно спрощує роботу з програмою навіть для користувачів, які не мають технічного досвіду.

На рисунку 3.1 наведено фрагмент реалізації графічного інтерфейсу програми у середовищі Visual Studio Code із використанням бібліотеки CustomTkinter.

Причини вибору Python як основної мови розробки:

– домінантність у галузі комп’ютерного зору та машинного навчання;

– простий, читабельний синтаксис, що дозволяє зосередитися на логіці алгоритмів;

– широке розмаїття бібліотек з відкритим кодом;

– активна спільнота та велика кількість прикладів;

– можливість швидкого створення прототипів і гнучкість налаштувань.

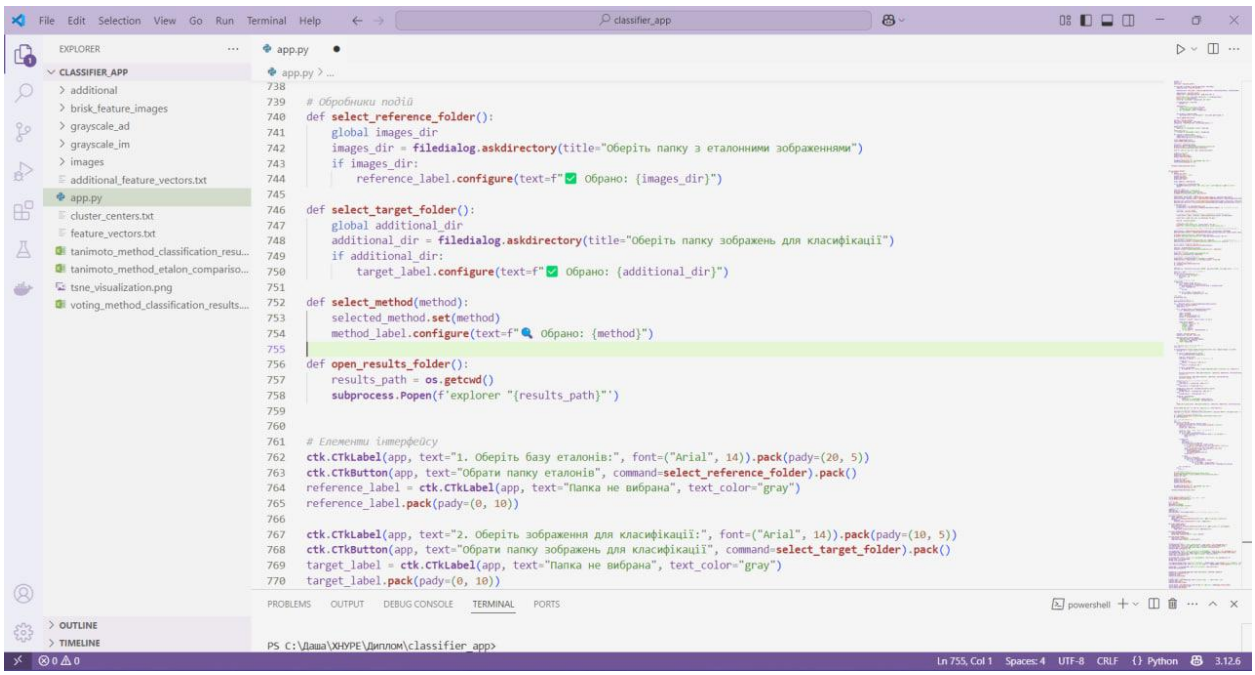


Рисунок 3.1 – Реалізація обробників подій та інтерфейсу користувача в середовищі VS Code

Таким чином, використання Python та сучасних бібліотек дозволило ефективно реалізувати поставлені задачі для застосунку для класифікації зображень.

3.2 Особливості програмної реалізації

Програма реалізує класифікацію зображень двома способами: методом голосування та методом із використання метрики Танімото. Початок роботи обох методів однаковий. Еталонні і вхідні зображення завантажуються в методі load_and_process_images(), і там переводяться у напівтоновий вигляд. Це робиться тому, що BRISK колірна інформація не потрібна, так як він працює саме з градаціями сірого.

Коли всі зображення підготовлено, кожне з них подається на вхід детектору BRISK, який знаходить на зображенні ключові точки і створює їх дескриптори. Із них програма обирає лише 500 найкращих – тих, які мають

найвище значення параметру *response*. Дескриптори, які спочатку є у стиснутому вигляді, розпаковуються в повноцінні послідовності нулів і одиниць, тобто як бінарні вектори. Вони зберігаються у списку – кожен набір відповідає одному зображенню.

Далі при виборі методу голосування у методі classify_image(), визначається, на які з відомих класів або еталонних зображень найбільше схоже вхідне зображення.

Процес починається з того, що створюється словник *votes*, у якому кожному з відомих еталонів (тобто підписаних зображень, що вже були проаналізовані раніше) призначається нульова кількість голосів. Це буде своєрідне голосування: кожен дескриптор з вхідного зображення шукає найближчий до себе дескриптор серед усіх еталонів і «віддає голос» за той, який виявиться найближчим за відстанню Геммінга.

Далі для кожного дескриптора з вхідного зображення метод переглядає всі дескриптори з кожного еталонного зображення і обирає той, який має найменшу відстань Геммінга. Якщо ця мінімальна відстань не перевищує заданого порогу для встановлення еквівалентності пари дескрипторів, то голос цього дескриптора віддається за відповідне еталонне зображення, тобто його клас.

Після того як усі дескриптори з тестового зображення «проголосували», система підраховує найбільшу кількість голосів. Якщо є еталон, який отримав більше голосів, ніж заданий поріг для вирішального значення максимуму числа голосів класів, то саме він оголошується переможцем – тобто, вважається, що вхідне зображення належить до цього класу. Якщо ж жоден клас не набрав достатньої кількості голосів, то програма повертає результат «Не визначено». Отримані результати зберігаються в таблиці у файлі Excel.

Якщо обрати спосіб класифікації методом з використанням метрики Танімото, то спочатку еталони проходять всю необхідну попередню обробку, а вже потім – вхідні зображення.

Таким чином, усі виділені 500 дескрипторів кожного еталону додаються до загального списку *descriptor_list*, а два найсильніші дескриптори з кожного зображення витягуються окремо та зберігаються в списку *initial_centers*. Потім ці дані передаються в метод kmeans_hamming().

У методі kmeans_hamming() реалізується варіація алгоритму *k*-середніх, але замість звичайної евклідової відстані між точками використовується відстань Геммінга, яка підходить для кластеризації бінарних векторів. Програмний вигляд функції вказаний у лістингу 3.1.

Лістинг 3.1 Реалізація методу кластеризації алгоритмом *k*-середніх з використанням відстані Геммінга:

```
def kmeans_hamming(initial_centers, X, max_iter=500):
    centroids = initial_centers
    n_clusters = len(centroids)

    for _ in range(max_iter):
        distances = hamming_distance_matrix(X, centroids)
        labels = np.argmin(distances, axis=1)
        new_centroids = np.zeros_like(centroids)

        for i in range(n_clusters):
            cluster_points = X[labels == i]
            if len(cluster_points) > 0:
                new_centroids[i] = np.round(np.mean(cluster_points, axis=0)).astype(int)

        if np.all(new_centroids == centroids):
            break

        centroids = new_centroids
```

return centroids, labels

Робота функції починається з того, що задаються початкові центроїди кластерів, які передаються у вигляді параметра *initial_centers*. Кількість кластерів визначається за довжиною цього масиву центрів.

Алгоритм далі виконується циклічно, щонайбільше *max_iter* разів. На кожній ітерації обчислюється матриця відстаней Геммінга – це відстань між кожною точкою (тобто бінарним вектором у масиві) і кожним центроїдом. Після цього кожна точка отримує мітку – відноситься до того кластеру, чий центр найближчий. Мітки зберігаються у масиві *labels*.

Далі формується новий набір центрів кластерів. Для кожного кластера збираються всі точки, які до нього належать, і обчислюється новий центр: для кожної координати вектора береться середнє значення серед усіх точок у кластері. Оскільки робота відбувається з бінарними векторами, ці середні значення округлюються до 0 або 1 і отримується новий бінарний вектор, що найкраще представляє групу. Коли центри перестають змінюватися або після досягнення максимального числа ітерацій функція повертає фінальні центри кластерів і мітки кластерів для кожної точки.

Додатково у методі visualize_tsne() виконується двовимірна візуалізація отриманих дескрипторів ознак після їх попередньої кластеризації. Замість стандартної евклідової відстані t-SNE тут також налаштовано використання відстані Геммінга. Після трансформації дані стають придатними для відображення на площині у вигляді розсіювання точок. У візуалізації кожна точка відповідає одному вхідному дескриптору, а її колір визначається номером кластера, до якого вона належить відповідно до отриманиих міток *labels*. Додатково переданий масив *centroids*, тобто координати центрів кластерів, також проєктується і предсталений на графіку.

Повертаючись до основного алгоритму, набори дескрипторів зображень і центри метод compute_feature_vectors(). Він використовується для того, щоб

перетворити набір бінарних дескрипторів зображень у компактні вектори ознак для кожного з них. Програмний вигляд функції вказаний у лістингу 3.2.

Лістинг 3.2 Реалізація методу побудови векторів ознак:

```
def compute_feature_vectors(descriptor_list, cluster_centers, delta):
    cluster_assignments = []
    for descriptors in descriptor_list:
        feature_vector = np.zeros(len(cluster_centers), dtype=int)
        distances = hamming_distance_matrix(descriptors, cluster_centers) # (n_descriptors, n_clusters)
        min_distances = np.min(distances, axis=1)
        valid_indices = np.where(min_distances <= delta)[0]
        cluster_labels = np.argmin(distances[valid_indices], axis=1) if len(valid_indices) > 0 else []
        for label in cluster_labels:
            feature_vector[label] += 1
        cluster_assignments.append(feature_vector)
    return np.array(cluster_assignments)
```

Робота відбувається з набором *descriptor_list*, де кожен елемент – це масив дескрипторів з одного зображення. Для кожного такого масиву створюється вектор ознак – нульовий масив довжиною, що дорівнює кількості кластерів. Далі для кожного дескриптора в зображенні обчислюється його відстань Геммінга до всіх центрів кластерів. Після цього визначається, який центр є найближчим до кожного дескриптора, і якщо мінімальна відстань до нього не перевищує заданий допустимий поріг, дескриптор зараховується як такий, що підходить. Для таких дескрипторів визначається найближчий центр, і відповідний лічильник у векторі ознак збільшується.

Таким чином, кожен дескриптор або враховується у векторі ознак зображення, або ігнорується, якщо його відстань до всіх центрів занадто

велика. У підсумку, для кожного зображення формується вектор входжень у кластери, який відображає, які з кластерів були найбільш представлені серед його локальних ознак. Метод повертає матрицю, де кожен рядок – це вектор ознак для одного зображення, придатний до подальшої класифікації.

Метод compute_tanimoto_metrics() використовується для попарного обчислення значень метрики Танімото між векторами ознак еталонів, поданими у вигляді масиву *feature_vectors*.

Для кожної пари векторів обчислюється такі величини:

- суми елементів у кожному векторі;
- перетин векторів;
- об’єднання векторів;
- симетрична різниця векторів;
- значення метрики Танімото.

Отримані результати зберігаються в таблиці у файлі Excel.

Переходячи до вхідних зображень, для них так само виділяються набори по 500 найкращих дескрипторів, які потім проєктуються на центри кластерів і для кожного формується свій вектор ознак. І далі, по аналогії з роботою метода compute_tanimoto_metrics(), для кожного вектора ознак вхідного зображення будується табличка з обчисленими значеннями параметрів необхідних для визначення метрики Танімото між ним та векторами ознак еталонів. Час витрачений на класифікацію кожного вхідного зображення засікається для подальшого визначення середнього і загального часу. Отримані результати зберігаються в табличках у файлі Excel.

3.3 Інструкція користувача

Для запуску застосунку користувачу необхідно мати встановлене середовище Python версії 3.8 або новішої. Якщо Python ще не інстальовано, його можна завантажити з офіційного сайту розробників. Після встановлення

потрібно підготувати робоче середовище, встановивши всі необхідні бібліотеки, які використовуються у програмі. Це можна зробити через термінал або командний рядок, скориставшись командою pip install, яка автоматично завантажить і встановить усі потрібні залежності.

Серед сторонніх бібліотек, які необхідно встановити окремо, використовуються:

– opencv-python – для обробки зображень;

– numpy – для числових обчислень;

– pandas – для роботи з табличними даними;

– matplotlib – для побудови графіків;

– customtkinter – для створення сучасного графічного інтерфейсу;

– scikit-learn – зокрема для реалізації алгоритму t-SNE;

– openpyxl – для зчитування та запису Excel-файлів.

Окрім цього, у програмі застосовуються також модулі, які входять до стандартної бібліотеки Python і не потребують окремого встановлення. До них належать:

– os, time, threading, subprocess – для взаємодії з операційною системою та багатопоточності;

– tkinter, включаючи filedialog і messagebox – для реалізації віконного інтерфейсу.

Після завершення встановлення всіх залежностей користувач може запустити програму безпосередньо з Python, відкривши основний скрипт у середовищі розробки або виконавши його через командний рядок. У результаті відкриється графічне вікно інтерфейсу, де можна взаємодіяти з програмою – обирати зображення, аналізувати їх та переглядати результати обробки.

Після запуску програми користувач побачить головне вікно графічного інтерфейсу, створене за допомогою бібліотеки customtkinter. Вікно має простий, інтуїтивно зрозумілий дизайн і містить усі необхідні елементи для взаємодії із застосунком. Інтерфейс складається з інструкцій у вигляді

підписів, кнопок для вибору папок, інструментів запуску алгоритмів класифікації та відображення статусу виконання завдань (рис. 3.2).

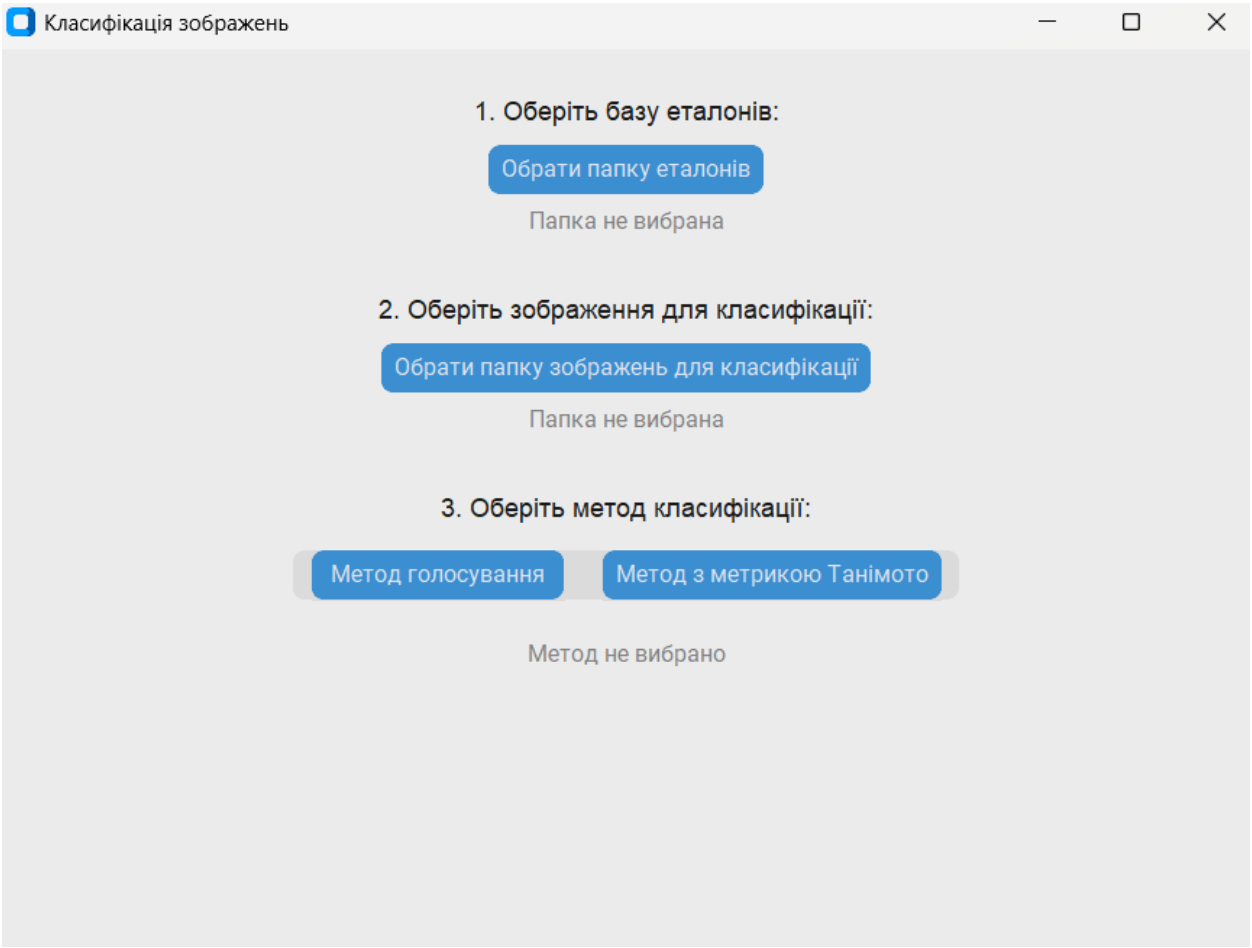


Рисунок 3.2 – Головне вікно програми після запуску

Першим кроком є вибір папки з еталонними зображеннями. Це зображення, які попередньо класифіковані вручну та використовуються як база для порівняння з іншими зображеннями. Натиснувши кнопку «Обрати папку еталонів», відкриється стандартне діалогове вікно, де користувач може вказати відповідну директорію. Після вибору шляху програма відобразить підтвердження під кнопкою: текст зміниться повний шлях до обраної папки, що свідчить про успішне збереження шляху.

На другому етапі користувачу потрібно обрати папку зображень, які необхідно класифікувати. Це ті зображення, клас яких ще невідомий, і які програма повинна порівняти з еталонною базою, аби визначити найбільш

ймовірний клас для кожного з них. Як і в попередньому випадку, після натискання кнопки «Обрати папку зображень для класифікації» відкривається вікно вибору, а після підтвердження – в інтерфейсі з’явиться текст із шляхом до вибраної папки.

Коли обидві папки обрано, користувач може перейти до вибору методу класифікації. Для цього передбачено дві окремі кнопки:

– «Метод голосування» – виконує класифікацію на основі підрахунку голосів еталонних зображень, які мають найбільшу подібність до вхідного зображення за дескрипторами;

– «Метод з метрикою Танімото» – використовує коефіцієнт Танімото для оцінки схожості між зображеннями.

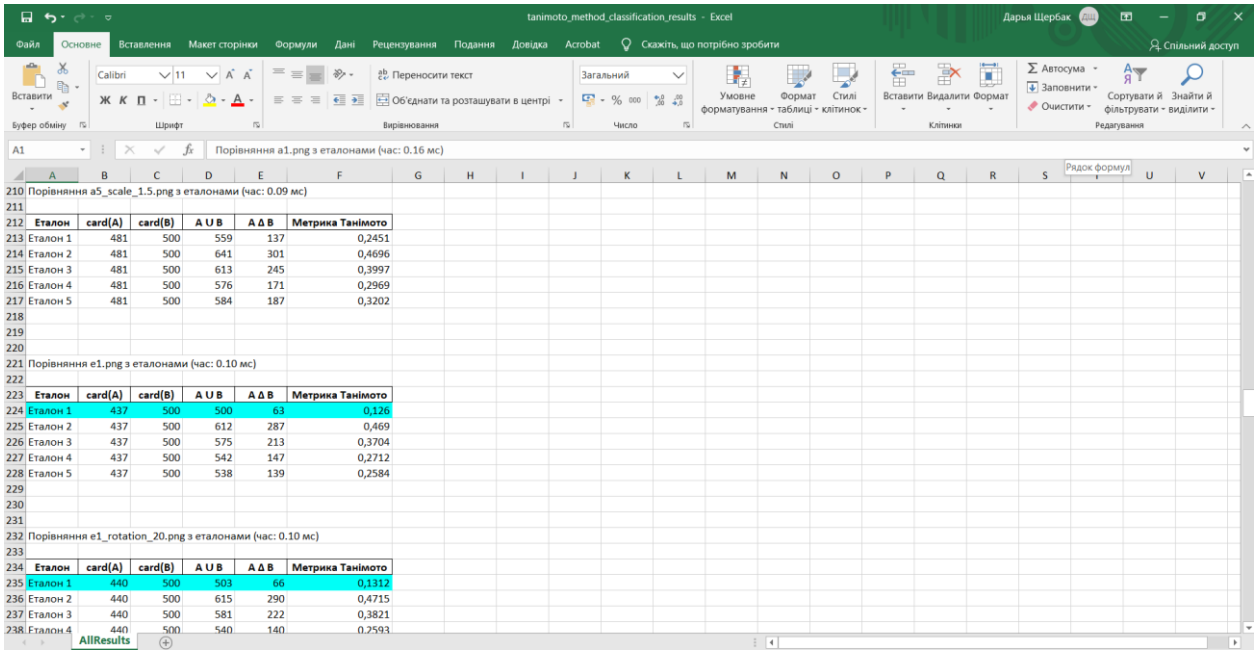
Після вибору будь-якого методу програма автоматично запускає процес обробки. При цьому з’являється індикатор прогресу – горизонтальна смуга, яка починає рухатись у режимі «очікування», сигналізуючи про активну роботу алгоритму. Нижче відображається текст «Обробка зображень…», який додатково інформує користувача про те, що програма зайнята класифікацією.

У разі вибору методу голосування програма може працювати повільніше, особливо при великій кількості зображень, оскільки кожне з них порівнюється з усіма еталонами за дескрипторами. Натомість метод з метрикою Танімото зазвичай працює швидше, адже він працює з векторами ознак і виконує менше обчислень.

Коли класифікацію завершено, індикатор прогресу зникає, а натомість з’являється кнопка «Відкрити результати». Ця кнопка відкриває папку, де збережено результати роботи. Тут містяться файли з результатами роботи програми, наприклад файли Excel з інформацією про класифікацію вхідних зображень.

Файл із результатами класифікації зберігається у форматі .xlsx та відкривається в табличному редакторі, наприклад, Microsoft Excel. Усі результати згруповано за зображеннями, які проходили класифікацію. Кожна така група містить назву зображення, час, витрачений на його обробку, а також

таблицю з результатами порівняння з еталонами. Для кожного еталона вказано кількість дескрипторів у зображенні, кількість дескрипторів в еталоні, об’єднання множин, симетричну різницю та значення метрики Танімото, яка характеризує ступінь подібності між зображенням і кожним еталоном. Еталон, який отримав найменше значення метрики Танімото, автоматично вважається найбільш схожим. На рисунку 3.3 видно, що найменше значення метрики виділено кольором, що зручно для подальшого аналізу.



Порівняння a5_scale_1.5.png з еталонами (час: 0.09 мс)

Еталон	card(A)	card(B)	A ∪ B	A Δ B	Метрика Танімото
Еталон 1	481	500	559	137	0,2451
Еталон 2	481	500	641	301	0,4696
Еталон 3	481	500	613	245	0,3997
Еталон 4	481	500	576	171	0,2969
Еталон 5	481	500	584	187	0,3202

Порівняння e1.png з еталонами (час: 0.10 мс)

Еталон	card(A)	card(B)	A ∪ B	A Δ B	Метрика Танімото
Еталон 1	437	500	500	63	0,126
Еталон 2	437	500	612	287	0,469
Еталон 3	437	500	575	213	0,3704
Еталон 4	437	500	542	147	0,2712
Еталон 5	437	500	538	139	0,2584

Порівняння e1_rotation_20.png з еталонами (час: 0.10 мс)

Еталон	card(A)	card(B)	A ∪ B	A Δ B	Метрика Танімото
Еталон 1	440	500	503	66	0,1312
Еталон 2	440	500	615	290	0,4715
Еталон 3	440	500	581	222	0,3821
Еталон 4	440	500	540	140	0,2593

Рисунок 3.3 – Приклад результатів класифікації зображень за метрикою Танімото у форматі Excel

3.4 Тестування розробленої моделі

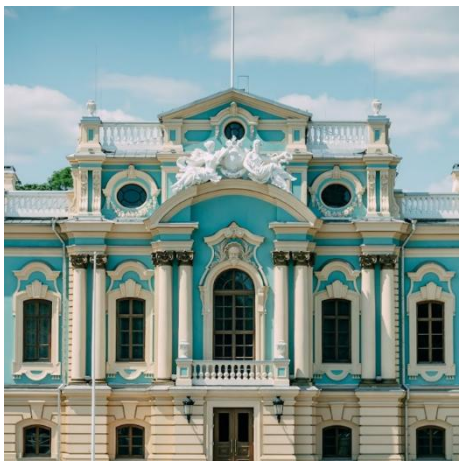
Для роботи обрано десять зображень об’єктів в різних архітектурних стилях [43]. З них п’ять зображень сформували базу еталонів (рис. 3.4), а решта п’ять – вибірку, яка не входить до бази (рис. 3.5). Бароко, готика, метаболізм, модернізм, класицизм, індо-ісламський (могольський) стиль, ренесанс, раджастханський стиль – усі ці стилі мають виразні характерні риси, що дозволяє виділяти велику кількість ключових точок для подальшого аналізу.

(а)

(б)

(в)

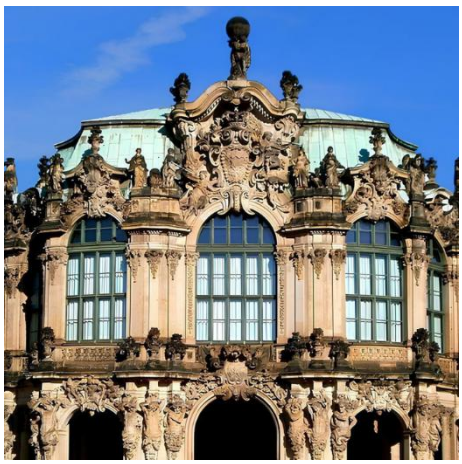
(г)



(ґ)

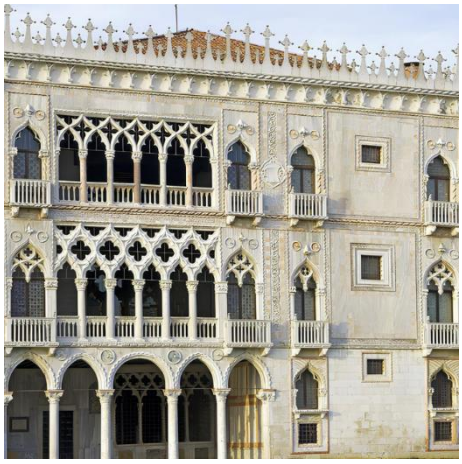
Рисунок 3.4 – Зображення із бази еталонів: (а) центр культури в Медельїні, Колумбія; (б) палац в Джайпурі, Індія; (в) собор в Йорку, Англія; (г) Накагінська капсульна вежа, Японія; (ґ) Маріїнський палац, Україна

(а)

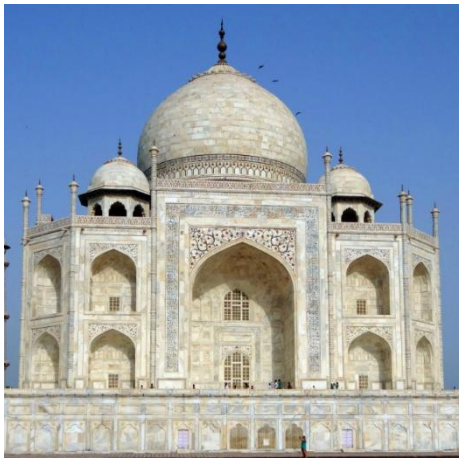


(б)

(в)



(г)

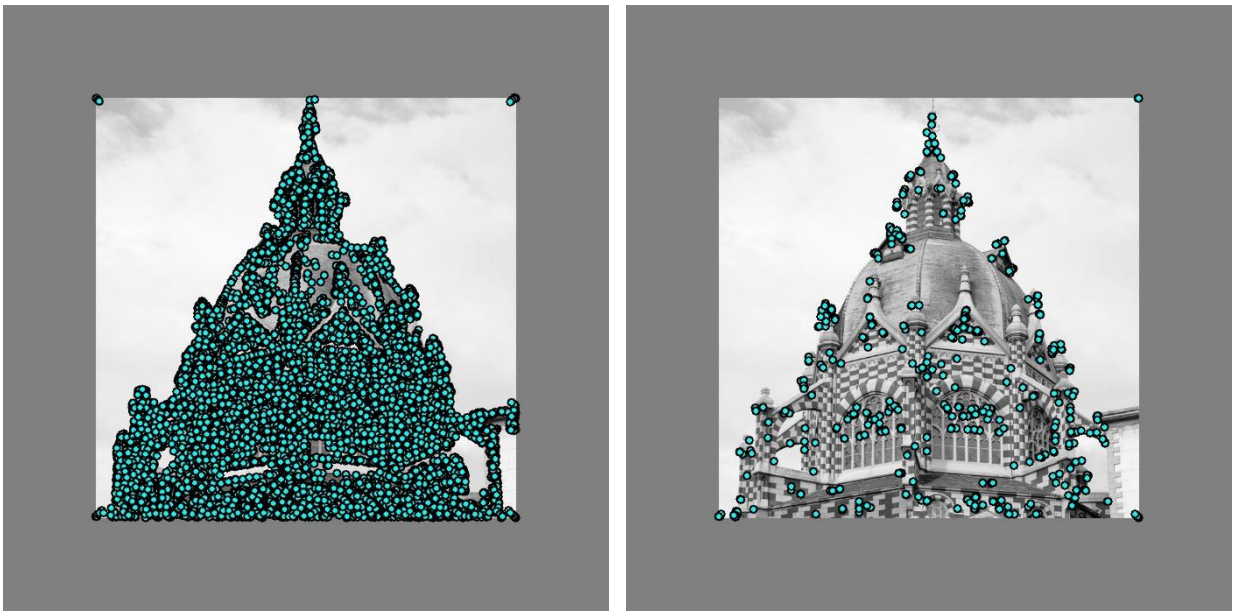


(г)

Рисунок 3.5 – Зображення поза базою: (а) церква у Флоренції, Італія; (б) місце проведення заходів в Дрездені, Німеччина; (в) медичний центр в штаті Невада, США; (г) палац Ка’д’Оро у Венеції, Італія; (г) мавзолей в Агрі, Індія

На етапі підготовки усі зображення було переведено у напівтоновий вигляд, обрізано до розміру 700×700 пікселів і розміщено по центру на сірому фоні розміром 1100×1100 пікселів. Таким чином ми уникнемо зміщення об’єктів відносно рамок зображення при трансформаціях і забезпечимо однакові умови для виявлення ключових точок.

Далі для кожного зображення було виділено ключові точки методом BRISK, відсортовано їх за параметром *response* та відібрано по 500 найбільш значущих із них. Це дозволить нам скоротити обсяг обчислень і зосередитися лише на найбільш характерних особливостях зображення. На рисунку 3.10 (а) представлено приклад із бази еталонних зображень із виділеними на ньому дескрипторами усіх ключових точок, а на рисунку 3.10 (б) – позначені лише 500 відібраних точок.



(а) (б)

Рисунок 3.6 – Приклад зображення з виділеними координатами ключових точок: (а) усі знайдені дескриптори; (б) виділені 500 дескрипторів відсортованих за параметром *response*

Над отриманою множиною дескрипторів була виконано кластеризація методом *k*-середніх із використанням відстані Геммінга, у результаті чого було сформовано десять центрів кластерів.

На рисунку 3.7 представлено візуалізацію кластерів і їх центрів за допомогою алгоритму t-SNE для демонстрації даних у двовимірному вигляді, використовуючи відстань Геммінга. Кожна точка на графіку відповідає дескриптору, а кольори відображають приналежність до певного кластера. Центри кластерів також зображено на графіку відповідними кольорами, тож можна наочно побачити розподіл дескрипторів у проєктованому просторі.

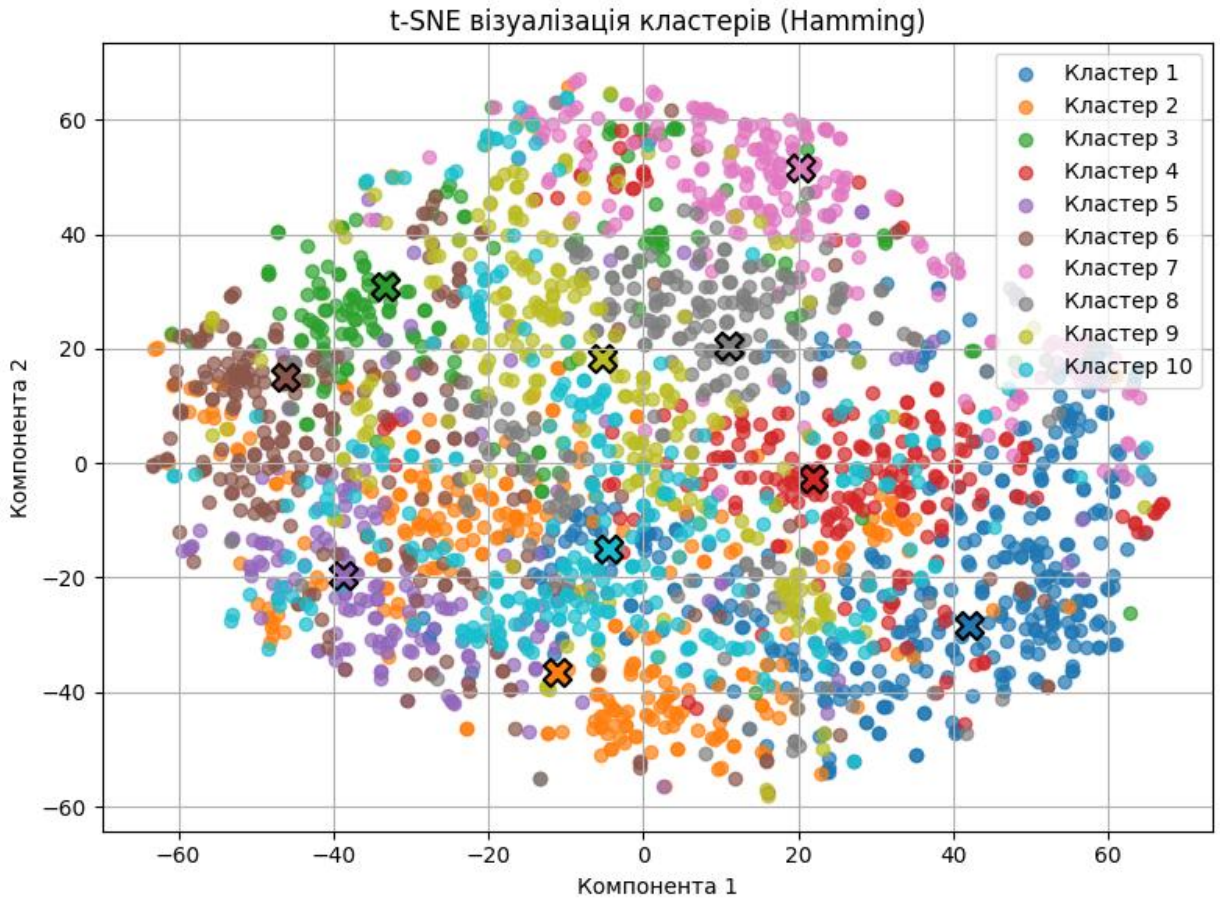


Рисунок 3.7 – Візуалізація кластерів і їх центрів за допомогою алгоритму t-SNE з використанням відстані Геммінга

Такий підхід дозволяє не лише впорядкувати дескриптори за схожістю, але й виявити приховані структури у вхідних даних. Використання методу

t-SNE, зокрема, дає змогу інтуїтивно оцінити щільність і компактність кластерів, а також просторову відстань між ними, що може свідчити про ступінь відмінності між різними типами ознак.

Після завершення кластеризації було сформовано стислий опис кожного зображення у вигляді вектора ознак. Для цього на отримані центри проєктуємо набори дескрипторів з кожного об’єкту та підраховуємо кількість дескрипторів, що були віднесені до кожного із центрів. Таким чином для кожного зображення сформовано вектор із десяти чисел, який узагальнено описує структуру зображення у просторі кластеризованих дескрипторів.

У таблиці 3.1 представлено отримані вектори ознак для зображень із бази еталонів.

Таблиця 3.1 – Вектори ознак для еталонних зображень

Номер еталону	Кількість «голосів» для центрів									
	1	2	3	4	5	6	7	8	9	10
1	101	54	14	45	36	29	47	62	41	71
2	54	26	63	26	23	41	76	74	66	51
3	90	96	21	51	35	76	24	31	36	40
4	69	57	42	58	28	41	31	48	68	58
5	76	52	21	38	52	71	44	40	36	70

Для класифікації зображень об’єктів було застосовано два підходи: метод голосування та метод з використанням кластерного подання, заснований на метриці Танімото. Метою експерименту є оцінка здатності системи правильно розпізнавати зразки, що належать до відомих класів (еталонні зображення), а також не пропускати незнайомі об’єкти (зображення, що не входять до бази), не відносячи їх помилково до жодного з класів.

У таблиці 3.2 наведено результати класифікації еталонних зображень. Як видно, усі 5 зразків були правильно класифіковані, причому кожне з них отримало по 500 голосів за свій клас і по 0 – за інші.

Таблиця 3.2 – Результати класифікації еталонних зображень методом голосування

Номер еталонного зображення	Кількість «голосів» за клас					Визначений клас
	1	2	3	4	5	
1	**500**	0	0	0	0	1
2	0	**500**	0	0	0	2
3	0	0	**500**	0	0	3
4	0	0	0	**500**	0	4
5	0	0	0	0	**500**	5

У таблиці 3.3 наведено результати класифікації зображень поза базою. Жодне з них не подолало встановлений поріг у 250 голосів, тобто належність до будь-якого з класів не була визначена.

Таблиця 3.3 – Результати класифікації зображень поза базою методом голосування

Номер зображення поза базою	Кількість «голосів» за клас					Визначений клас
	1	2	3	4	5	
1	21	31	47	32	27	Не визначено
2	63	38	68	82	42	Не визначено
3	127	33	44	92	112	Не визначено
4	58	35	86	39	84	Не визначено
5	70	25	21	40	56	Не визначено

Метод голосування, як і очікувалося, виявився точним, але досить повільним. Для нього було задано поріг еквівалентності пари дескрипторів, що дорівнює 128, а також поріг для вирішального значення кількості голосів класів – 250, що є стандартними значеннями для цього методу. Аналіз використання і характеристики порогів наведено у розділі 2. Як видно із таблиць 3.2 і 3.3, усі еталонні зображення були правильно класифіковані, тоді як зображення поза базою не набрали достатньої кількості голосів, щоб подолати поріг і бути віднесеними до будь-якого з класів. Крім цього, середній

час, витрачений методом голосування на класифікацію одного зображення, склав 9380 мілісекунд.

Метод, заснований на метриці Танімото, було протестовано у трьох варіантах, що відрізняються значеннями порогів для відстані між дескрипторами об’єктів і центрами кластерів.

Під час порівняння еталонів між собою використовується додатково поріг фільтрації даних для відстані між аналізованим дескриптором і центром кластера, що впливає на формування векторів ознак. Далі, при класифікації вхідних зображень, для кожного з них обчислюється вектор ознак аналогічним чином, але з можливим іншим порогом. Потім для кожного вхідного зображення обчислюється метрика Танімото.

Були протестовані такі три варіанти налаштувань порогів:

– обидва пороги – і для еталонів, і для вхідних зображень дорівнюють 512 (фільтрація даних відсутня);

– обидва пороги дорівнюють 192 (описи еталонів і вхідних зображень скорочуються під дією фільтрації за відстанню до центрів);

– для еталонів поріг дорівнює 512 (описи еталонів використовуються у повному об’ємі), а для вхідних зображень – 192 (фільтрація здійснюється для дескрипторів аналізованих зображень).

Поріг 192 – це приблизно 37,5% від повного опису об’єкта (512), що було обрано емпірично шляхом методу проб і помилок. При менших значеннях порогу, наприклад 128 (тобто 25% від повного опису), надто багато дескрипторів відсікається, що призводить до втрати важливої інформації та неможливості правильної класифікації зображень. Натомість при більших значеннях, близьких до 512, ефект фільтрації стає надто слабким, що зменшує контрастність між подібними та непов’язаними зображеннями. Тому значення 192 було визначено як компромісне.

Для перших двох варіантів налаштувань порогів система правильно класифікує всі еталони, але деякі із об’єктів поза базою все ж проходять поріг подібності та бувають помилково класифіковані.

Третій варіант з використанням повних еталонів і фільтрованих описів вхідних зображень показав себе найкраще. У таблиці 3.4 наведено результати попарного порівняння повних еталонів між собою, де жирним шрифтом позначено мінімальне значення метрики Танімото. Далі при класифікації вхідних зображень це значення використовується як поріг для міри подібності зображень.

Таблиця 3.4 – Результати попарного порівняння повних описів еталонів

Еталон A	Еталон B	Кількість елементів A	Кількість елементів B	Кількість елементів об’єднання (A ∪ B)	Кількість різних елементів (A Δ B)	Значення метрики Танімото
1	1	500	500	500	0	0
1	2	500	500	627	254	0,4051
1	3	500	500	602	204	0,3389
1	4	500	500	583	166	0,2847
1	5	500	500	565	130	**0,2301**
2	2	500	500	500	0	0
2	3	500	500	678	356	0,5251
2	4	500	500	592	184	0,3108
2	5	500	500	638	276	0,4326
3	3	500	500	500	0	0
3	4	500	500	602	204	0,3389
3	5	500	500	576	152	0,2639
4	4	500	500	500	0	0
4	5	500	500	586	172	0,2935
5	5	500	500	500	0	0

У таблицях 3.5 і 3.6 приведено результати класифікації вхідних зображень з використанням третього варіанту оброблення: видно, що всі еталони були правильно розпізнані, а об’єкти поза базою не були віднесені до жодного класу. Середній час класифікації склав 0,10 мілісекунд. Це у 94 рази швидше, ніж методом голосування. У подальшому використовується саме цей варіант налаштувань порогів для методу з використанням метрики Танімото.

Таблиця 3.5 – Результати класифікації вхідних еталонних зображень методом з використанням метрики Танімото

Номер вхідного зображення	Номер порівнюваного еталону	Кількість елементів у вхідному зображенні	Кількість елементів у еталоні	Значення метрики Танімото	Визначений клас
1	1	417	500	**0,166**	1
	2	417	500	0,4405	
	3	417	500	0,3327	
	4	417	500	0,2763	
	5	417	500	0,2567	
2	1	445	500	0,401	2
	2	445	500	**0,11**	
	3	445	500	0,5257	
	4	445	500	0,3392	
	5	445	500	0,438	
3	1	450	500	0,3953	3
	2	450	500	0,5294	
	3	450	500	**0,1**	
	4	450	500	0,3705	
	5	450	500	0,2601	
4	1	473	500	0,296	4
	2	473	500	0,3224	
	3	473	500	0,3224	
	4	473	500	**0,054**	
	5	473	500	0,3049	
5	1	475	500	0,2682	5
	2	475	500	0,4325	
	3	475	500	0,2774	
	4	475	500	0,2984	
	5	475	500	**0,05**	

Таблиця 3.6 – Результати класифікації вхідних зображень поза базою методом з використанням метрики Танімото

Номер вхідного зображення	Номер порівнюваного еталону	Кількість елементів у А	Кількість елементів у В	Значення метрики Танімото	Визначений клас
1	**2**	**3**	**4**	**5**	**6**
1	1	305	500	0,4519	Не визначено
	2	305	500	0,4811	
	3	305	500	0,4925	
	4	305	500	0,4725	
	5	305	500	0,4459	

Продовження таблиці 3.6

1	2	3	4	5	6
2	1	448	500	0,3162	Не визначено
	2	448	500	0,5047	
	3	448	500	0,3542	
	4	448	500	0,3627	
	5	448	500	0,3711	
3	1	488	500	0,3024	Не визначено
	2	488	500	0,4823	
	3	488	500	0,3169	
	4	488	500	0,3803	
	5	488	500	0,2787	
4	1	466	500	0,3141	Не визначено
	2	466	500	0,3846	
	3	466	500	0,3112	
	4	466	500	0,3258	
	5	466	500	0,2781	
5	1	417	500	0,3297	Не визначено
	2	417	500	0,2955	
	3	417	500	0,4588	
	4	417	500	0,3683	
	5	417	500	0,377	

Додатково проведено серію експериментів із вхідними зображеннями (еталонними та зображеннями поза базою) із такими комбінаціями трансформацій:

– поворот на кут ±20°;

– зміна масштабу до 0,9 і 1,1.

Для кожного зображення було створено 5 варіацій: одне оригінальне (без змін) і чотири трансформовані версії. Таким чином, при загальній кількості 10 початкових зображень було сформовано 50 вхідних зображень для тестування.

Ці експерименти проводяться з метою перевірки стійкості системи до простих афінних перетворень, які часто трапляються у реальних умовах – зокрема, через варіативність ракурсів зйомки або відстані до об’єкта. На рисунку 3.8 показані приклади зображень з перетвореннями повороту і масштабування.

Рисунок 3.8 – Приклади перетворень повороту і масштабування

Порівняння результатів показало, що обидва підходи – як голосування, так і метрика Танімото – продемонстрували успішну класифікацію вхідних зображень у межах експериментальної вибірки.

Однак слід зазначити, що для методу з використанням метрики Танімото результати значною мірою залежать від характеристик підібраного набору зображень. У подальшому було виявлено, що перестановка ролей між еталонними зображеннями та зображеннями поза базою (тобто зміна їх місцями) призводить до появи помилкових класифікацій. Це свідчить про чутливість запропонованого підходу до формування еталонної бази та про необхідність ретельного добору репрезентативних прикладів.

Натомість класичне голосування продемонструвало стабільну роботу навіть за змінених умов. Це узгоджується з його статусом перевіреного підходу, що вже зарекомендував себе у низці подібних задач. Відповідно, для розроблюваного методу з метрикою Танімото варто провести додаткові дослідження, зокрема щодо впливу якості дескрипторів, особливостей нормалізації та фільтрації ознак, а також масштабування на більших наборах зображень.

У таблиці 3.7 представлено результати порівняння двох підходів до класифікації: методу голосування та методу з використанням метрики Танімото. Зокрема, оцінювалася точність і загальний час класифікації всіх вхідних зображень.

Таблиця 3.7 – Точність і час оброблення для методів класифікації

Метод	Точність	Середній час класифікації, мс
Голосування	1	9380
Танімото	1	0,10

Результати свідчать, що метод голосування забезпечує високу точність класифікації, однак потребує значного часу на оброблення – у середньому понад 9 секунд на одне зображення. Такий час зумовлений необхідністю повного перебору дескрипторів усіх еталонних зображень.

Водночас метод із використанням коефіцієнта Танімото демонструє суттєво кращу швидкодію – лише 0,1 мілісекунди на класифікацію одного зображення. У наведених умовах він забезпечив таку саму точність, проте подальші експерименти показали залежність результатів від складу еталонної бази.

Отже, метод голосування варто використовувати там, де першочергове значення має точність і допустимий тривалий час обробки – наприклад, при невеликому обсязі даних або високих вимогах до надійності. Метод із метрикою Танімото, завдяки високій швидкодії, краще підходить для систем реального часу та великих баз. Проте його чутливість до вибору еталонів потребує додаткових досліджень щодо формування еталонної бази та перевірки результатів. У майбутньому доцільно розглянути поєднання обох підходів: Танімото – для швидкої попередньої оцінки, голосування – для остаточного рішення.

ВИСНОВКИ

У межах кваліфікаційної роботи було розроблено, реалізовано та досліджено два підходи до класифікації зображень: метод голосування та метод, заснований на використанні метрики Танімото. Обидва методи працюють з попередньо сформованими наборами дескрипторів і можуть бути застосовані для розв’язання задач аналізу візуальної інформації.

Дослідження показали, що методи відрізняються як у швидкодії, так і в стійкості до зміни складу еталонної бази. Метод голосування забезпечує високу точність навіть за змінених умов, проте має суттєві часові витрати, що зростають зі збільшенням кількості еталонів. Натомість метод із метрикою Танімото демонструє надзвичайно високу швидкість класифікації та стабільну масштабованість, але є чутливим до вибору еталонів, що може впливати на точність за певних змін.

Таким чином, метод голосування доцільно використовувати в задачах із пріоритетом точності, а метод Танімото – у системах реального часу або при роботі з великими обсягами даних. Перспективним напрямом є комбінування цих підходів для поєднання їхніх сильних сторін.

У подальшому доцільно дослідити масштабування методів на великі вибірки, перевірку їх стійкості до шумів та адаптацію до задач із неповною або спотвореною інформацією.

Результати роботи апробовано у формі тез доповідей під час Міжнародного молодіжного форуму «РАДІОЕЛЕКТРОНІКА І МОЛОДЬ У XXI СТОЛІТТІ» [44].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Sinha, S., Shukla, R., & Rastogi, E. (2023). Computer vision applications and challenges: A review. International Research Journal of Modernization in Engineering Technology and Science, 05(12), 3404-3414.

2. Yadav, P., Singh, H., & Khanna, K. (2022, February). Computer Vision, Its Applications, and Challenges. In Proceedings of the International Conference on Innovative Computing & Communication (ICICC).

3. Understanding computer vision – Part 1.
URL: https://bayoftech.medium.com/understanding-computer-vision-part-1-1f5c5b67a12d (дата звернення 03.04.2025).

4. Bhalani, V. (2024). Advancements in computer vision: Techniques, applications, and future trends. Journal of Information and Computational Science, 14(12).

5. Ivanov, A. (2025). A comprehensive review of the history and methods of computer vision. Telekomunikatsiini ta informatsiini tekhnolohii, 86(1), 176-191.

6. Amato, G., Carrara, F., Ciampi, L., Di Benedetto, M., Gennaro, C., Falchi, F., Messina, N., & Vairo, C. (2022, September). AI and computer vision for smart cities. Paper presented at I-Cities 2022, Ascoli Piceno, Italy. Institute of Information Science and Technologies – National Research Council.

7. Tymchyshyn, R., Volkov, O., Hospodarchuk, O., & Bohachuk, Yu. (2018). Сучасні підходи до розв’язання задач комп’ютерного зору [Modern approaches to solving computer vision problems]. Control Systems and Computers, (6), 46-73.

8. 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 (Eds.), Explainable Deep Learning Artificial Intelligence (pp. 125-146). Academic Press.

9. Göktepe, Y. E., & Uzun, Y. (2024). Computer vision: Object recognition and classification. In S. Koçer & Ö. Dündar (Eds.), Intelligent systems and optimization in engineering (pp. 163-171). ISRES Publishing.

10. Hassaballah, M., Alshazly, H. A., & Ali, A. A. (2019). Analysis and evaluation of keypoint descriptors for image matching. In M. Hassaballah & K. M. Hosny (Eds.), Recent advances in computer vision (pp. 79-113). Springer.

11. Gorokhovatskyi, V., Chmutov, Y., Tvoroshenko, I., & Kobylin, O. (2025). Reducing computational costs by compressing the structural description in image classification methods. Advanced Information Systems, 9(1), 5-12.

12. Gorokhovatskyi, V., Tvoroshenko, I., Yakovleva, O., & Hudáková, M. (2025). Image description compression in classification structural methods. IEEE Access, 13, 43631-43641.

13. Gorokhovatskyi, V., Gadetska, S., & Stiahlyk, N. (2024). Classification of images based on distance assessment. In V. Snytyuk (Ed.), Information Technology and Implementation (Satellite): Conference Proceedings (pp. 22-24).

14. Gorokhovatskyi, V. O., Gadetska, S. V., & Stiahlyk, N. I. (2019). [Вивчення статистичних властивостей моделі блочного подання для множини дескрипторів ключових точок зображень]. Радіоелектроніка, інформатика, управління, (2), 100-107.

15. Gorokhovatsky, V. A., & Putyatin, Y. P. (2008). Structural recognition of images on the basis of voting models of attributes of typical points. Data Recording, Storage and Processing, 10(4), 75-85.

16. Gorokhovatskyi, V. A., & Zamula, A. A. (2016). Employment of intelligent technologies in multiparametric control systems. Telecommunications and Radio Engineering, 75(19), 1775-1785.

17. 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.

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

19. Gorokhovatskyi, V., Gadetska, S., & Stiahlyk, N. (2023). Accelerating image classification based on a model for estimating descriptor-to-class distance. International Journal of Computing, 22(4), 485-492.

20. Gorokhovatskyi, V., Tvoroshenko, I., Yakovleva, O., Hudáková, M., & Gorokhovatskyi, O. (2024). Application of a committee of Kohonen neural networks to training of image classifier based on description of descriptors set. IEEE Access, 12, 73376-73385.

21. Gorokhovatskyi, V., & Tvoroshenko, I. (2024). An effective method for transforming an image description into a compact vector for classification. In V. Snytyuk (Ed.), Information Technology and Implementation (Satellite): Conference Proceedings (pp. 25-28). Publishing House «Caravela».

22. Pupchenko, D., & Gorokhovatskyi, V. (2024). Accelerated filtration of ultrasound images. In V. Snytyuk (Ed.), Information Technology and Implementation (Satellite): Conference Proceedings (pp. 69-72). Publishing House «Caravela».

23. Nguyen, T. T., Krishnakumari, P., Calvert, S. C., Vu, H. L., & van Lint, H. (2019). Feature extraction and clustering analysis of highway congestion. Transportation Research Part C: Emerging Technologies, 100, 238-258.

24. Awad, A. I., & Hassaballah, M. (Eds.). (2016). Image feature detectors and descriptors: Foundations and applications (Vol. 630). Springer.

25. Binary Descriptors.
URL: https://www.epfl.ch/labs/cvlab/research/descriptors-and-keypoints/research-detect-binarydescriptors/ (дата звернення 17.05.2025).

26. Alshazly, H. A., Hassaballah, M., Ali, A. A., & Wang, G. (2018). An experimental evaluation of binary feature descriptors. In A. E. Hassanien, A. Taha, K. Shaalan, & M. Azar (Eds.), Proceedings of the International Conference on

Advanced Intelligent Systems and Informatics 2017 (Vol. 639, pp. 181-191). Springer.

27. Moghimi, A., Celik, T., Mohammadzadeh, A., & Kusetogullari, H. (in press). Comparison of keypoint detectors and descriptors for relative radiometric normalization of bitemporal remote sensing images. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing.

28. Leutenegger, S., Chli, M., & Siegwart, R. Y. (2011, November). BRISK: Binary Robust invariant scalable keypoints. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (pp. 2548-2555). IEEE.

29. BRISK tutorial.
URL: https://juliaimages.org/ImageFeatures.jl/v0.0.3/tutorials/brisk.html (дата звернення 04.04.2025).

30. cv::BRISK class reference.
URL: https://docs.opencv.org/4.x/de/dbf/classcv_1_1BRISK.html (дата звернення 06.04.2025).

31. cv::KeyPoint class reference.
URL: https://docs.opencv.org/4.x/d2/d29/classcv_1_1KeyPoint.html (дата звернення 10.04.2025).

32. Alkhawlani, M., Elmogy, M., & Elbakry, H. (2015). Content-based image retrieval using local features descriptors and bag-of-visual words. International Journal of Advanced Computer Science and Applications, 6(9), 212-219.

33. Bag of visual words. URL: https://www.pinecone.io/learn/series/image-search/bag-of-visual-words/ (дата звернення 12.04.2025).

34. Gorokhovatskyi, V., Tvoroshenko, I., & Yakovleva, O. (2024). Transforming image descriptions as a set of descriptors to construct classification features. Indonesian Journal of Electrical Engineering and Computer Science, 33(1), 113-125.

35. Gorokhovatsky, V., Putyatin, Y., & Stolyarov, V. (2017). Research of effectiveness of structural image classification methods using cluster data model. Radio Electronics, Computer Science, Control, 3(42), 78-85.

36. Gorokhovatskyi, V., Tvoroshenko, I., & Yakovleva, O. (2024). Transforming image descriptions as a set of descriptors to construct classification features. Indonesian Journal of Electrical Engineering and Computer Science, 33(1), 113-125.

37. Гороховатський, В. О., & Гадецька, С. В. (2020). Статистичне оброблення та аналіз даних у структурних методах класифікації зображень (монографія). Харків: ФОП Панов А. Н.

38. Python & machine learning. URL: https://www.netguru.com/blog/python-machine-learning (дата звернення 11.05.2025).

39. Why Python is the leader in artificial intelligence and data science. URL: https://vishwajeetsinghrana8.medium.com/why-python-is-the-leader-in-artificial-intelligence-and-data-science-57ef9f8dfc88 (дата звернення 14.05.2025).

40. cv::BRISK class reference.
URL: https://docs.opencv.org/4.x/de/dbf/classcv_1_1BRISK.html (дата звернення 05.04.2025).

41. From Tkinter to CustomTkinter: A journey to better UIs in Python. URL: https://medium.com/@VaishnavSabariGirish/from-tkinter-to-customtkinter-a-journey-to-better-uis-in-python-fd9ea6388fda (дата звернення 13.04.2025).

42. CustomTkinter documentation.
URL: https://customtkinter.tomschimansky.com/documentation/ (дата звернення 15.05.2025).

43. Tripadvisor. URL: https://www.tripadvisor.com/ (дата звернення 17.05.2025).

44. Щербак Д.Д. (2025) Квантування ознак у методах класифікації зображень. Радіоелектроніка і молодь у XXI столітті: тези доповідей 29-го Міжнародного молодіжного форуму (Харків, 16–19 квітня 2025 р.). Харків: ХНУРЕ, 2025. Т. 7. С. 182-184.