Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

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

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ВИВЧЕННЯ МЕТОДУ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ НА ПІДСТАВІ ОРТОГОНАЛЬНОГО РОЗКЛАДАННЯ ДАНИХ
(тема)

Виконав:
студент 2 курсу, групи ІНФм-21-1
Чмутов Ю. В.
(прізвище, ініціали)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

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

Допускається до захисту

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

2022 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

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

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Чмутову Юрію Вадимовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Вивчення методу класифікації зображень на підставі ортогонального розкладання даних

затверджена наказом по університету від 9 листопада 2022 року № 1469Ст

2. Термін подання студентом роботи до екзаменаційної комісії 21 листопада 2022 р.

3. Вихідні дані до роботи детектування ключових точок на зображенні, структурні методи класифікації зображень за допомогою детекторів ключових точок, ортогональні перетворення.

4. Перелік питань, що потрібно опрацювати в роботі
1. Аналіз існуючих структурних методів класифікації зображень.
2. Апарат розкладання компонентів опису за системою ортогональних функцій.
3. Моделювання стиснення простору ознак.
4. Програмна реалізація та аналіз результатів.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) результати роботи методів класифікації зображень, виділені дескриптори ORB, база еталонних зображень.

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

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

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	09.11.2022	
2	Аналіз завдання, підбір літератури	09.11.22-11.11.22	
3	Аналіз літератури з досліджуваної проблеми	12.11.22-14.11.22	
4	Аналіз технічних засобів	15.11.22-16.11.22	
5	Розробка методу	17.11.22-19.11.22	
6	Програмна реалізація	20.11.22-21.11.22	
7	Оформлення пояснювальної записки	22.11.22-23.11.22	
8	Перевірка на плагіат		
9	Рецензування		
10	Підготовка презентації та доповіді		
11	Занесення роботи в електронний архів		
12	Попередній захист кваліфікаційної роботи		

Дата видачі завдання 9 листопада 2022 р.

Студент _____
(підпис)

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

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 68 с., 3 табл., 26 рис., 1 дод., 46 джерел.

КОМП'ЮТЕРНИЙ ЗІР, СТРУКТУРНІ МЕТОДИ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ, ФУНКЦІЇ УОЛША, ШВИДКОДІЯ ОБРОБЛЕННЯ, РЕЗУЛЬТАТИВНІСТЬ КЛАСИФІКАЦІЇ.

Об'єктом дослідження є структурні методи класифікації зображень.

Метою дослідження є удосконалення результативності структурних методів класифікації зображень шляхом впровадження апарату розкладання компонентів опису за системою ортогональних функцій Уолша та застосування моделей стиснення простору ознак.

Використано методи: апарат теорії множин, метричні моделі для визначення релевантності щодо множин багатовимірних векторів, теорія ортогонального розкладення векторів. Проведено дослідження швидкодії та ефективності розроблених методів класифікації багатовимірних даних. Експерименти підтвердили суттєве скорочення обчислювальних витрат у розроблених модифікаціях класифікаторів на основі впровадження ортогонального розкладення даних.

У результаті дослідження здійснена програмна реалізація методів класифікації, отримано експериментальні показники їх функціонування.

COMPUTER VISION; STRUCTURAL METHODS OF IMAGE CLASSIFICATION; ORB DESCRIPTOR; ORTHOGONAL DECOMPOSITION; WALSH FUNCTIONS; PROCESSING SPEED; EFFECTIVENESS OF CLASSIFICATION.

The object of research is structural methods of image classification.

The aim of the research is to improve the effectiveness of structural methods of image classification by implementing the apparatus for decomposition of description components according to the system of orthogonal functions of Walsh and the use of feature space compression models.

Methods used: set theory apparatus, metric models for determining relevance to sets of multidimensional vectors, theory of orthogonal decomposition of vectors. A study of the speed and efficiency of the developed methods of classification of multidimensional data was conducted. Experiments confirmed a significant reduction of computing costs in the developed modifications of classifiers based on the implementation of orthogonal data decomposition.

As a result of the research, implementation of classification methods was carried out, experimental indicators of their functioning were obtained.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ФУ – функції Уолша

СКВ – середньоквадратичне відхилення

КТ – ключові точки

OpenCV – Open Source Computer Vision Library

ВСТУП

Одним із ключових завдань теорії розпізнавання даних є формування продуктивної системи інформативних ознак задля забезпечення результативної засобів класифікації зображень у комп'ютерному зорі [1-5]. Найбільш пріоритетним напрямком новітніх досліджень є вивчення природи аналізованих даних з метою встановлення в їх складі наявних чи скритих закономірностей, наприклад, спільних чи відмінних характеристик, кластерів тощо [6, 7], і впровадження отриманих знань у процес розпізнавання [8-10].

Паралельно до цього розвивається інша група методів. Це методи, у яких аналіз базується на проектуванні довільних вхідних даних на фіксований простір ортогональних функцій, а також на використанні значень отриманих проекцій як ознак для розпізнавання. Дані методи носять більш формалізований характер. У відповідності до встановленої процедури, вони здатні універсально опрацьовувати будь-які дані у заданому просторі. У залежності від ступеня розподільної здатності вибраної системи функцій стосовно аналізованих даних змінюється ефективність цих підходів [11].

Часто є можливість дещо скоротити повну множину використовуваних функцій при опрацюванні ортогональних систем стосовно заданої бази ознак зображень. Це робиться для створення підмножини, що задовольняє потребам результативної класифікації. У такому випадку пропорційно ступеню скорочення та за рахунок стиснення модифікованого простору ознак також вдається досягти суттєвого зменшення обчислювальних затрат.

Для методів класифікації на підставі розкладання, базова множина для проектування аналізованих описів зображень уже задана. Тому її не треба формувати у процесі навчання класифікатора. Технологія навчання тут може полягати як у адаптації до еталонної інформації, так і у формуванні ефективної підмножини, яка дає можливість у заданий часовий діапазон здійснювати результативну класифікацію [12, 13].

У структурних методах класифікації образ об’єкта подається скінченною множиною векторів, що є дескрипторами ключових точок (КТ) зображення [14, 15]. Тоді варіант ефективного подання може бути пов’язаний із перетворенням образу, тобто наявної множини дескрипторів, до ортогонального простору. Прикладний інтерес через достатньо просту програмну та апаратну реалізації має застосування ортогональної системи функцій Уолша (ФУ) та перетворень, пов'язаних з ними [16]. Впровадження кусково-постійних базисних функцій Уолша характеризується незначними обчислювальними затратами, так як не пов’язано з діями множення і реалізується у просторі цілих чисел.

Система функцій Уолша (Адамара) дає можливість здійснити розкладання вхідного сигналу (окремо кожного дескриптора КТ) за сімейством прямокутних базисних функцій. Отримана множина цілочисельних векторів-ознак є ґрунтовною підставою для побудови ефективного класифікатора.

Актуальність дослідження полягає в удосконаленні структурних методів класифікації зображень шляхом впровадження апарату розкладання компонентів опису за системою ортогональних функцій та застосуванні моделей стиснення простору ознак задля пришвидшення класифікації.

1 ФОРМУВАННЯ ПРОСТОРУ ОЗНАК У МЕТОДАХ КЛАСИФІКАЦІЇ ЗОБРАЖЕНЬ

1.1 Формалізація задачі класифікації візуальних образів

Розпізнавання образів – це розділ теорії штучного інтелекту. Даний розділ вивчає методи класифікації об’єктів. Під образом мається на увазі деяка упорядкована сукупність ознак, наприклад, дескрипторів КТ. Класом образів (або просто класами) називається сукупність образів, які мають деякі спільні властивості. У ролі образу можуть бути цифрові фотографії, символи, цифри, запис мови (розпізнавання мови) тощо. У рамках теорії штучного інтелекту розпізнавання образів узагальнює нову наукову дисципліну – теорію машинного навчання, метою якої є розроблення методів та алгоритмів, які мають можливість здійснювати навчання [17].

Зазвичай, прийнято використовувати термін «класифікація» замість звичного «розпізнавання». Дані терміни є синонімічними, але вони не є повністю взаємозамінними. Кожен з них застосовується у конкретних сферах, тому їх розуміння та вживання часто залежить від специфіки конкретної задачі.

Як і кожна математична дисципліна, розпізнавання образів також має власний математичний апарат. Він включає геометрію та алгебру, дискретну математику, методи оптимізації, апарат штучного інтелекту. Класифікація образів має широке застосування при створенні комп’ютерних систем, на які покладаються інтелектуальні функції. Це ті функції, які пов’язані з прийняттям інтелектуальних рішень замість людини. Наприклад, це може бути пошук інформації, інтелектуальний аналіз даних, медична діагностика або криміналістична експертиза.

У практичних задачах поширились три форми упорядкованого представлення ознак: у вигляді векторів ознак (для кількісних дескрипторів), у вигляді символьних рядків, а також у вигляді дерев (рядки та дерева

застосовуються для структурних описів). Образи, що представлені векторами ознак, мають форму

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \tag{1.1}$$

де кожна з компонент x_i представляє *i*-ий дескриптор, а n – це загальна кількість дескрипторів, що пов'язані з даним образом. Образи представляються у вигляді вектор-стовпця (тобто матрицями порядку $n \times 1$) або у еквівалентній формі $x = \left(x_1, x_2, \ldots, x_n\right)^T$, де T – це операція транспонування.

Опис у формі символьних рядків дає можливість породжувати адекватні образи для таких об'єктів, у яких структура базується на відносно нескладних способах поєднання примітивів. Такий зв'язок, зазвичай, відповідає формам границь фігур. Для багатьох прикладних задач застосовується більш потужний підхід, який базується на описі за допомогою дерев. Більшість ієрархічно упорядкованих схем приводять до деревовидних структур. Наприклад, на рисунку 1.1 представлено супутникове зображення щільно забудованої центральної частини міста та навколишніх житлових передмість. Наведене на рисунку 1.2 представлення зображення у вигляді дерева, яке обернене коренем вгору, було отримано за допомогою структурного відношення «складається з». Корінь дерева представляє зображення повністю. Наступний рівень показує, що зображення складено з центральної частини міста та житлової зони, яка, у свою чергу, складається з житлових будинків, шосейних доріг та площ з магазинами. Наступний рівень пропонує подальшу деталізацію опису для житлових будівель та шосейних

доріг. Розбиття подібного виду може продовжуватись аж до межі нашої властивості розмежовувати відмінної області на зображенні.

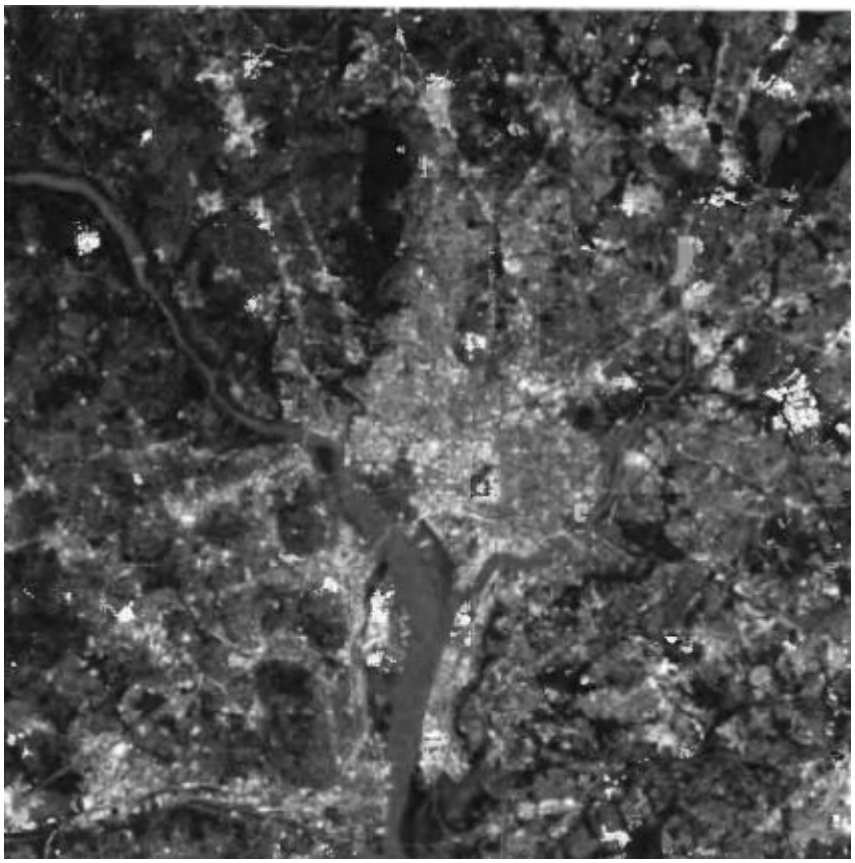


Рисунок 1.1 – Супутникове зображення центру м. Вашингтон

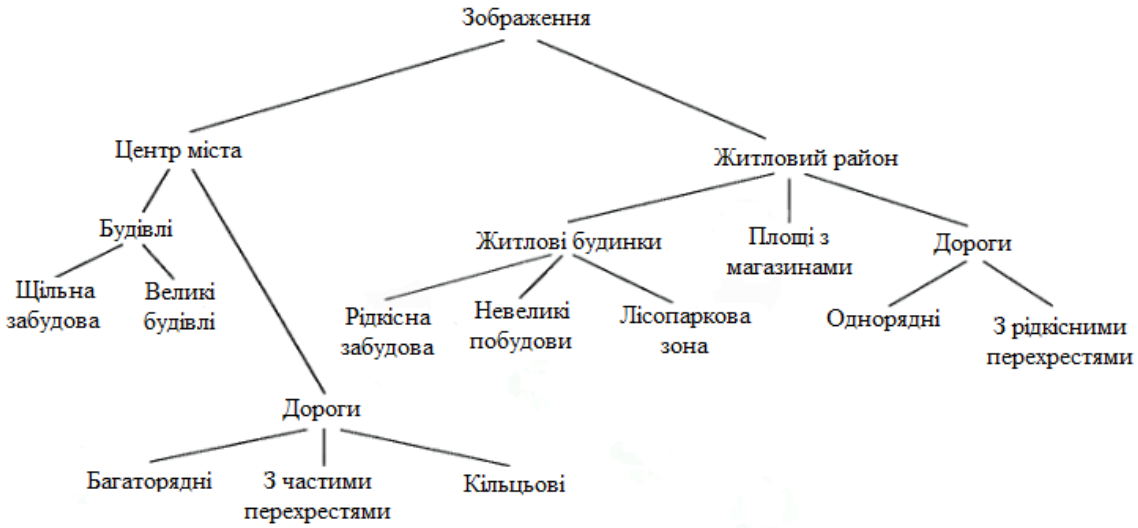


Рисунок 1.2 – Опис рисунку 1.1 у вигляді дерева

Сформулюємо математичну постановку задачі класифікації [18].

Нехай Ω – простір образів; $\omega \in \Omega$ – образ; $M = \{1,2,...,m\}$ – номери класів $\Omega_1, \Omega_2, ..., \Omega_m$, таких, що $\Omega_i \cap \Omega_j = \varnothing$, $\varnothing$ – пуста множина, та $\Omega = \bigcup_{i=1}^{m} \Omega_i$, $g : \Omega \to M$ – індикаторна функція, що є невідомою; X – простір ознак, тобто векторний простір, точками якого є вектори ознак образів; $x : \Omega \to X$ – функція, що ставить у відповідність образу ω його вектор ознак $X(\varpi\}$, $K_1, K_2, ..., K_m$ – підмножини простору X, такі що $K_i \cap K_j = \varnothing$ та $X = \bigcup_{i=1}^{m} K_i$, $g : X \to M$ – вирішальне правило, яке ставить у відповідність вектору ознак образа номер класу, якому він належить. Побудова вирішального правила – це мета розпізнавання.

Ідея задачі класифікації з учителем – на підставі навчальної вибірки, яка представлена у вигляді множини прецедентів (g_j, x_j), $j = 1,...,N$, мати можливість створити вирішальне правило g – за його допомогою здійснюється мінімізація кількості помилок. У ролі вчителя виступає сама навчальна вибірка, або той, ким було вказане значення g_j.

Відповідно, задача класифікації без вчителя зазвичай називається кластеризацією або просто кластерним аналізом. У даній задачі вибірка ознак образів x_j, $j = 1,...,N$ розбивається на підмножини. Ці множини називаються кластерами і вони не перетинаються. Кластери складаються зі схожих один на одного об’єктів. Важливо, щоб об’єкти, що належать до різних кластерів, вагомо відрізнялись один від одного.

Приклади типових постановок задач класифікації [19]:

– віднесення об’єкта до конкретного класу. Це можуть бути задачі розпізнавання символів у тексті. Або задачі, у рамках яких необхідно визначити чи присутній якийсь дефект у технічній деталі. Зазвичай, говорячи про розпізнавання образів, найчастіше мають на увазі проблему віднесення об’єкта до певного класу;

– кластерний аналіз. Мета такого аналізу полягає у розподілі заданого набору об’єктів на класи. Як приклад, це можуть бути дані про кліматичні умови у різних регіонах. Клас це група об’єктів, які мають деяку схожість між собою за різними критеріями та показниками. Дана задача називається класифікацією без вчителя тому що класи апріорно не задані;

– задача ідентифікації. Метою даної задачі є виокремлення конкретного об’єкту серед об’єктів, що подібні до нього. Прикладом є ситуація коли необхідно впізнати знайому людину серед натовпу.

Під час сприйняття просторової структури трьохвимірного оточуючого світу у зоровій системі людини враховуються ознаки, що відповідають за дальність (або ознаки глибини об’єктів) декількох типів.

Задача класифікації з формального огляду полягає у наступному [20].

Маємо K класів, які позначатимемо S_1, $S_2, \dots S_K$. Задано множину прецедентів – об’єкти, для кожного з яких відомо, до якого з цих класів він належить:

$$X=\{X_i,y_i;\ i=\overline{1,N}\}=\{(x_{i,1}\ x_{i,2}\ \dots\ x_{i,p}),y_i;i=\overline{1,N}\}=\begin{pmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,p} & y_1 \\ x_{2,1} & x_{2,2} & \dots & x_{2,p} & y_2 \\ \dots & \dots & \dots & \dots & \dots \\ x_{N,1} & x_{N,2} & \dots & x_{N,p} & y_N \end{pmatrix}, \qquad (1.2)$$

де $X_i = (x_{i,1}\ x_{i,2}\ \dots\ x_{i,p})$ – i-й об’єкт, описаний p ознаками;

$x_{i,j}$ – значення j-ї ознаки для i-го об’єкта;

y_i – назва класу, до якого належить i-й об’єкт;

N – кількість об’єктів;

p – кількість ознак.

Ця множина називається навчальною вибіркою.

На основі цієї вибірки необхідно побудувати таке правило, яке б давало можливість відносити будь-який новий об’єкт $X_0 = (x_{0,1}\ x_{0,2}\ \dots\ x_{0,p})$ до одного з класів $S_1, S_2, \dots S_K$.

Задача розв’язується двома етапами:

Етап 1. У першу чергу – виконується навчання**.** На етапі навчання будується вирішальне правило. Цей етап, як правило, складний і більш трудомісткий. Геометрично на етапі навчання будується поверхня, яка розділяє класи. Однак, не завжди можливо визначити аналітичний вигляд цієї поверхні.

Етап 2. Тепер черга класифікації. На основі побудованого правила здійснюється класифікація нового об’єкта.

Одним з найпоширеніших способів класифікації даних є метод на основі порівняння з еталонами – узагальненими описами класу. Суть методу полягає у тому, що необхідно побудувати на етапі навчання еталони кожного класу з подальшою класифікацією нових об’єктів за правилом найближчого або k найближчих еталонів [21, 22].

Існує декілька способів у залежності від опису еталонів класів.

Спосіб 1. У ролі еталонів можуть виступати один або декілька навчальних об’єктів. Саме вони забезпечують якість класифікації не гіршу, ніж за усією навчальною вибіркою. У свою чергу, задача навчання передбачає вибір еталонних об’єктів, наприклад, методом STOLP або FRiS-STOLP [23]. У результаті, навчальна вибірка може значно скоротитись за рахунок того, що з неї видаляються неінформативні та шумові об’єкти. За допомогою методу найближчого або k найближчих сусідів здійснюється класифікація нових об’єктів.

У випадку коли мають місце два класи, метод STOLP працює наступним чином [23]. Спочатку знаходять найбільш «напружені» примежові об’єкти – для кожного об’єкта обчислюється відстань до найближчого об’єкта свого класу (d_{in}) та найближчого об’єкта чужого класу (d_{out}). Відповідно, відношення $W=d_{in}/d_{out}$ характеризує величину ризику для даного об’єкта бути розпізнаним як об’єкт чужого класу. Серед об’єктів кожного класу обирається по одному з максимальним значенням величини W, які заносять до списку еталонів. Після цього, за правилом найближчого сусіда за

допомогою еталонів, виконується пробне розпізнавання всіх об’єктів навчальної вибірки. Серед тих об’єктів, що були класифіковані неправильно, обирають один із максимальним значенням *W*, і ним поповнюють список еталонів. Далі відбувається повторення пробного розпізнавання всіх об’єктів навчальної вибірки. Процедуру продовжується доти, поки не будуть розпізнаватися правильно всі навчальні об’єкти.

У випадку, коли кількість класів більша двох, описана процедура застосовується для кожної пари класів. Рекомендується розпочинати застосовувати метод для пари двох найближчих класів з метою прискорення роботи. Після того, як задача розв’язана, обирається наступна найближча пара класів. Вибір пари виконується доти, поки не будуть розглянуті всі пари. У випадку, якщо у ході розглядання поточної пари виявляється те, що для одного з класів, який входить до неї, на попередніх кроках вже було сформовано список еталонів, цей список включається із самого початку і за необхідності поповнюється. У якості відстані між класами доцільно застосовувати відстань між двома найближчими об’єктами двох різних класів (наприклад, відстань Хаусдорфа).

Спосіб 2. Усереднені об’єкти класу обираються у якості еталонів. Вони визначаються за формулою:

$$\overline{X}^{(l)} = \frac{1}{N_l} \sum_{X_i \in S_l} X_i, \; l = \overline{1, K}, \tag{1.3}$$

де N_l – кількість об’єктів у класі S_l.

Далі, за правилом найближчого сусіда класифікують новий об’єкт X_0:

$$X_0 \in S_l : d\left(X_0, \overline{X}^{(l)}\right) = \min_{h = \overline{1, K}} d\left(X_0, \overline{X}^{(h)}\right). \tag{1.4}$$

Способи формування еталонів не обмежуються цими підходами, їх розроблення продовжується.

1.2 Застосування ортогональних перетворень

Ортогональне перетворення – це лінійне перетворення $T: V \to V$, яке зберігає симетричний скалярний добуток. Зокрема, ортогональне перетворення (технічно, ортонормальне перетворення) зберігає довжини векторів та кути між векторами

$$< v, w \geq = < T\, v, T\, w >. \tag{1.5}$$

Важливою сферою застосування ортогональних перетворень є стиснення даних. Стиснення даних базується на теорії представлення сигналів, яка розглядає методи ефективного представлення сигналів певного класу або класів. У випадку, якщо дискретний сигнал містить *N* відліків, тоді його можна розглядати як точку *N*-вимірного простору. Тоді кожен відлік є координатою *N*-вимірного вектору даних *X*, який представляється у вигляді сигналу в цьому просторі. Для більшої ефективності представлення можна здійснити ортогональне перетворення *X*, що дає результат $Y = TX$, де *Y* та *T* – це вектор коефіцієнтів перетворення та матриця перетворення відповідно. Метою стиснення даних є можливість вибору підмножини M координат вектору *Y*, де *M* значно менший за *N*. Інші $N - M$ координати можна відкинути, при цьому не викликав значної помилки при відновленні сигналу по *M* координатам вектору *Y*. Відповідно, порівнювати ортогональні перетворення потрібно у відповідності з деякими критеріями помилок. Наприклад, одним з найбільш популярних критеріїв є критерій середньоквадратичної помилки.

Логічним наслідком наведеної інформації є стиснення даних, яке полягає у тому, що представлення сигналу можна використовувати для зменшення кількості надлишкової інформації.

Цікавою є ідея побудови оптимального ортогонального перетворення, оптимального у сенсі середньоквадратичного критерію.

Нехай *I* – це ортогональне перетворення, яке задане у вигляді

$$T' = \left[\varphi_1 \varphi_1 \ldots \varphi_N\right], \tag{1.6}$$

де φ – це *N*-вектори. Для більшої зручності базисні вектори $\{\varphi_m\}$ будемо вважати нормованими та ортогональними, тобто

$$\varphi'_i \, \varphi_j = \begin{cases} 1, & i = j; \\ 0, & i \neq j. \end{cases} \tag{1.7}$$

Для кожного вектору *X*, який належить до даного класу векторів вхідних даних, отримуємо

$$Y = TX, \tag{1.8}$$

де $X' = \left[x_1 x_2 \ldots x_N\right]$ та $Y' = \left[y_1 y_2 \ldots y_N\right]$. Із виразу (1.6) та (1.7) виходить, що $T' T = I$ та, відповідно, $X = T' Y = \left[\varphi_1 \varphi_2 \ldots \varphi_N\right] Y$, що можна записати у вигляді

$$X = y_1 \varphi_1 + y_2 \varphi_2 + \ldots + y_N \varphi_N = \sum_{i=1}^{N} y_i \varphi_i. \tag{1.9}$$

Бажано зберігати підмножину $\left\{y_1 y_2 \cdots y_M\right\}$ координат Y та при цьому отримати оцінку X. Це може бути реалізовано заміною інших $N - M$ координат, можна представити у вигляді $\Delta X = X - \tilde{X}(M)$, де ΔX – це вектор помилки, тобто

$$\Delta X = X - \sum_{i=1}^{M} y_i \varphi_i + \sum_{i=M+1}^{N} b_i \varphi_i, \tag{1.10}$$

де $\tilde{X}(M)$ означає оцінку X. Помилку, яка з'являється при відкиданні $N-M$ координат, можна представити у вигляді $\Delta X = X - \tilde{X}(M)$, де ΔX – це вектор помилки, тобто

$$\Delta X = X - \sum_{i=1}^{M} y_i \varphi_i - \sum_{i=M+1}^{N} b_i \varphi_i. \tag{1.11}$$

Із виразів (1.9) та (1.11) можна зробити висновок, що

$$\Delta X = \sum_{i=M+1}^{N} \left(y_i - b_i\right) \varphi_i. \tag{1.12}$$

Таким чином, середньоквадратична помилка $\varepsilon(M)$ визначається у вигляді

$$\varepsilon(M) = E\{||\Delta X||^2\} = E\{(\Delta X)'(\Delta X)\}. \tag{1.13}$$

Підстановка (1.12) у (1.13) приводить до

$$\varepsilon(M) = E\left\{ \sum_{i=M+1}^{N} \sum_{j=M+1}^{N} \left(y_i - b_i\right)\left(y_j - b_j\right) \varphi_i' \varphi_j \right\},$$

а в спрощеному вигляді даний запис має вигляд

$$\varepsilon(M) = \sum_{i=M+1}^{N} E\left\{ \left(y_i - b_i\right)^2 \right\}. \tag{1.14}$$

Із виразу (1.14) можна зробити висновок, що для кожного вибору φ_i та b_i отримуємо конкретне значення $\varphi(M)$. Необхідно визначити таку комбінацію цих величину для того, щоб мінімізувати $\varepsilon(M)$. Процедура обрання оптимальних φ_i та b_i розбивається відповідно на етапи.

Етап 1**.** Оптимальне значення b_i визначається із виразу

$$\frac{\partial}{\partial b_i} E\left\{\left(y_i - b_i\right)^2\right\} = -2\left[E\left\{y_i\right\} - b_i\right] = 0,$$

що у свою чергу призводить до

$$b_i = E\left\{y_i\right\}. \tag{1.15}$$

Далі із виразів (1.7) та (1.9) отримуємо

$$y_i = \varphi_i' X. \tag{1.16}$$

Таким чином, $b_i = \varphi_i' E\{\overline{X}\} = \varphi_i' \overline{X}$, де $\overline{X} = E\{X\}$. Так як різниця $\left(y_i - b_i\right)$ у (1.14) є скалярною величиною, тоді $\varepsilon(M)$ можна записати у вигляді

$$\varepsilon(M) = \sum_{i=M+1}^{N} E\left\{\left(y_i - b_i\right)\left(y_i - b_i\right)'\right\}. \tag{1.17}$$

Підстановка $y_i = \varphi_i' X$ та $b_i = \varphi_i' \overline{X}$ у виразі (1.17) призводить до

$$\varepsilon(M) = \sum_{i=M+1}^{N} \varphi_i' E\{(X - \overline{X})(X - \overline{X})'\}\varphi_i.$$

Оскільки $\sum_x = E\{(\overline{X} - X)(X - \overline{X})'\}$ є коваріаційною матрицею X, отримуємо

$$\varepsilon(M) = \sum_{i=M+1}^{N} \varphi_i' \sum_x \varphi_i. \tag{1.18}$$

Для знаходження оптимального φ_i необхідно не тільки мінімізувати $\varepsilon(M)$ по відношенню до φ_i, але також і задовільнити обмеження $\varphi_i'\varphi_i = 1$. При цьому користуються методом множників Лагранжа та мінімізують

$$\varepsilon(M) = \varepsilon(M) - \sum_{i=M+1}^{N} \beta_i \left[\varphi_i'\varphi_i - 1\right] = \sum_{i=M+1}^{N} \left\{\varphi_i' \sum_x \varphi_i - \beta_i \left[\varphi_i'\varphi_i - 1\right]\right\} \qquad (1.19)$$

по відношенню до φ_i, де β_i – множники Лагранжа. Можна показати, що

$$\nabla\varphi_i\left[\varphi_i' \sum_x \varphi_i\right] = 2\sum_x \varphi_i \text{ та } \nabla\varphi_i\left[\varphi_i'\varphi_i\right] = 2\varphi_i.$$

Таким чином, із виразу (1.19) можна отримати

$$\nabla\varphi_i[\varepsilon(M)] = 2\sum_x \varphi_i - 2\beta_i \varphi_i = 0,$$

що у свою чергу призводить до

$$\sum_x \varphi_i = \beta_i \varphi_i. \qquad (1.20)$$

За визначенням виразу (1.20) значить, що φ_i – це власний вектор коваріаційної матриці $\sum_x$, а β_i – відповідне *i*-е власне значення. Виражаючи β_i через λ_i та підставляючи (1.20) у (1.18), можна отримати мінімальне значення середньоквадратичної помилки у вигляді

$$\varepsilon_{min}(M) = \sum_{i=M+1}^{N} \lambda_i. \qquad (1.21)$$

Записане у виразі (1.9) є розкладання за власними векторами коваріаційної матриці. Дане розкладання має назву розкладання Карунена-Лоева. Вектори φ_i, які створюють T у (1.6), вони є власними векторами $\sum_x$.

Саме тому перетворення $Y = TX$ має назву перетворенням Карунена-Лоева (ПКЛ).

Задача мінімізації $\varepsilon(M)$ у літературі зі статистики має назву факторний аналіз або аналіз головних компонент. Можна зробити два висновки:

– перетворення Карунена-Лоева є оптимальним перетворенням для представлення сигналів по відношенню до критерію середньоквадратичної помилки;

– оскільки $Y = TX$, тоді визначення коваріаційної матриці $\sum_y$ буде мати наступний вигляд

$$\sum_y = T \sum_x T^{-1} = T \sum_x T'.$$

Оскільки T складається з власних векторів $\sum_x$, тоді

$$\sum_y = diag\left(\lambda_1, \lambda_2, \ldots, \lambda_N\right), \tag{1.22}$$

де λ_i, $i = 1, 2, \ldots, N$ – це власні значення $\sum_x$. Оскільки $\sum_y$ – це діагональна матриця, тоді маємо висновок, що координати вектору перетворених даних y_i у виразі (1.16) некорельовані.

Із виразу (1.21) можна зробити висновок, що ефективність коефіцієнту перетворення y_i для представлення вектору даних X визначається відповідним власним значенням. У випадку, якщо коефіцієнт y_k не враховується, тоді середньоквадратична помилка збільшується на відповідне власне значення λ'_k. Таким чином, необхідно обрати множину y_i, яка відповідає M найбільшим власним значенням, а усі інші y_i необхідно відкинути, оскільки їх можна замінити константами $b_i, i = M+1, \ldots, N$.

Оскільки власні значення є елементами $\sum_y$, які стоять на головній діагоналі, то вони відповідають дисперсіям коефіцієнтів перетворення $y_i, i = 1, 2, \ldots, N$. Для всіх інших перетворень $\sum_y$ містить ненульові

недіагональні елементи. Саме тому природнім критерієм під час обрання інформативної множини коефіцієнтів перетворення є збереження M коефіцієнтів з найбільшими дисперсіями, а усі інші $(N-M)$ коефіцієнти можна відкинути. Критерій вибору коефіцієнтів перетворення, що наведено вище, визначимо як дисперсійний критерій.

Графік дисперсій коефіцієнтів перетворення виступає у ролі графічного представленням дисперсійного критерію, де дисперсії знаходяться у порядку зменшення та нормовані до вигляду матриць $\sum_x$ або $\sum_y$. Нормування відбувається тому, що відношення дисперсії до суми дисперсій дає міру середньоквадратичної помилки, яка виникає під час відкидання φ_i у (1.19). Такий графік називають графіком розподілу дисперсії. На рисунку 1.3 наведено розподіл дисперсії, яке пов’язано з чотирьома різними перетвореннями. Площа, яка обмежена кожною кривою для заданого числа коефіцієнтів перетворення, є мірою енергії, яка міститься у цих коефіцієнтах. Загальна площа, яка обмежена кожною кривою, дорівнює одиниці у результаті нормування до суми дисперсій. Наприклад, при збереженні 20 коефіцієнтів з рисунку 1.3 можна зробити висновок, що перетворення упорядковується відповідно до ефективності наступним чином.

Перетворення 4 > перетворення 2 > перетворення 3 > перетворення 1.

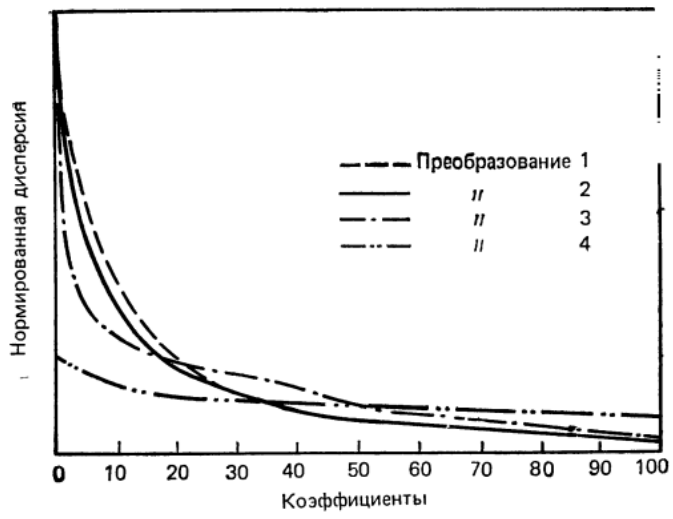


Рисунок 1.3 – Розподіл дисперсії

1.3 Постановка задачі дослідження

Таким чином, удосконалення результативності структурних методів класифікації зображень є актуальним завданням. Тому ставиться завдання дослідження та здійсненні ортогонального розкладання еталонних та вхідних образів, побудові метричного класифікатора у трансформованому просторі ознак, визначенні параметрів порогу для еквівалентності компонентів опису у різних метриках, вивчення результативності розроблених модифікацій класифікаторів шляхом програмного моделювання.

Об’єктом досліджень є структурні методи класифікації зображень.

Метою дослідження є удосконалення результативності структурних методів класифікації зображень шляхом впровадження апарату розкладання компонентів опису за системою ортогональних функцій Уолша та застосування моделей стиснення простору ознак.

Для досягнення мети необхідно вирішити такі завдання:

– провести аналіз існуючих структурних методів класифікації зображень;

– впровадити апарат розкладання компонентів опису за системою ортогональних функцій;

– реалізувати та застосувати моделі стиснення простору ознак;

– провести комп’ютерне тестування програмної моделі методу класифікації на прикладній базі зображень з порівнянням результатів традиційних та удосконалених методів класифікації;

– зробити висновки щодо проведеного дослідження.

2 МЕТОДИ КЛАСИФІКАЦІЇ ЗА СИСТЕМОЮ ОРТОГОНАЛЬНИХ ФУНКЦІЙ

2.1 Подання опису у ортогональному базисі

Опис $Z = \{z_i\}_{i=1}^{s}$ зображення у просторі структурних ознак представлено у вигляді скінченної множини, що складається з S векторів z_i, де $s = cardZ$, $z_i \in B^n$, $z_i = \{z_{i,j}\}_{j=1}^{n}$, а B^n – це векторний простір розмірності n з бінарними компонентами $\{0,1\}$. У свою чергу, у даній ситуації, дескриптор КТ зображення розглядається у вигляді точки n-мірного дискретного сигналу. Зважаючи на той факт, що у будь-якому скінченному векторному просторі може існувати хоча б одна ортогональна система векторів $W = \{\{w_{j,b}\}_{j=1}^{n}\}_{b=1}^{n}$. Ця система така, що будь-який вектор $z \in B^n, z = \{z_b\}_{b=1}^{n}$ може бути поданий у ній кортежем коефіцієнтів $\alpha = \{\alpha_j\}_{j=1}^{n}$, де

$$\alpha_j = (z \cdot w_j) = \sum_{b=1}^{n} z_b w_{j,b} . \qquad (2.1)$$

У свою чергу, кожен коефіцієнт (2.1) визначається у вигляді скалярного добутку вектору z на вектор w_j ортогональної системи. Коефіцієнти α_j мають назву координат вектору у базисі W. Для кожного $z_i \in Z$ виходить значення вектору розкладання α_i. У результаті впровадження математичної моделі векторного простору до класифікації виходить факт, що будь-який дескриптор $z_i \in Z$ образу зображення може бути обернено єдиним способом та подано у вигляді лінійної комбінації базисних векторів, а вибраний базис, за таких умов, складає точно n векторів, тобто

$$z_i = (\alpha_i \cdot w_j) = \sum_{b=1}^{n} \alpha_{i,b} w_{j,b}. \qquad (2.2)$$

Необхідно відмітити, що у n-мірному просторі векторів, теоретично, може існувати численна множина різних ортогональних базисів.

Орієнтуючись на той факт, що ФУ формально приймають значення, що має вигляд {1, –1} та безпосередньо не належить до векторного простору B^n, перетворення (2.1) відображає трансформацію $B^n \to C^n$ у простір C^n векторів, що мають цілі компоненти. У цей момент, простір $B^n \subset C^n$ є підпростором для простору C^n. Саме тому можна вважати, що перетворення (2.1), (2.2) здійснюються у просторі C^n. Коефіцієнт нормування для ФУ до ортонормальної системи, який забезпечує зворотність перетворень (2.1), (2.2), є константою при заданому n. З цієї причини, при обчисленнях, цей коефіцієнт можна не враховувати.

У процесі використання поелементної трансформації $z_i \to \alpha_i$ опис Z має вигляд $\alpha(Z) = \{\alpha_i\}_{i=1}^{s}$. Це означає, що множина $\{z_i\}_{i=1}^{s}$, $z_i \in B^n$ дескрипторів КТ перетворюється у множину векторів $\{\alpha_i\}_{i=1}^{s}$, $\alpha_i \in C^n$, яка має таку саму розмірність та чисельність. Множину α_i називають розподіленням потужності сигналу z_i. Виникає питання про результативність здійснення такої трансформації для подання даних у новому просторі ознак при здійсненні класифікації зображень.

Векторні простори B^n, C^n відносять до евклідових, де для будь-якого вектора z можна ввести норму

$$||z|| = \sqrt{(z \cdot z)} = \sqrt{\sum_{j=1}^{n} z_j z_j}, \qquad (2.3)$$

а наявність поняття скалярного добутку дає можливість визначити кут між векторами y, z як

$$\cos\varphi = \frac{(y \cdot z)}{||y|| \cdot ||z||}. \qquad (2.4)$$

У свою чергу, вираз (2.4) наряду із визначенням метрики $\varrho(y,z)$ для векторів доволі часто використовується у системах, що розпізнають дані. Але, при цьому, необхідно, щоб виконувалась умова $|\cos\varphi| \le 1$.

Квадрат норми (2.3) у просторі B^n відповідає числу одиничних бітів.

Також перетворення (2.1), (2.2) можуть бути записані у векторній формі. Пропонується застосувати розкладання опису у базисі сімейства прямокутних базисних функцій – дискретних ФУ. Вони є векторами цілих чисел (складається зі значень, що дорівнюють +1 або –1) скінченної розмірності. Дана розмірність, зазвичай, дорівнює ступеню двійки. Повний набір ФУ розмірності n утворює ортогональну матрицю Адамара A, яка має розміри $n \times n$. Дана матриця складається із n векторів ФУ $w_1,,w_n$. Наприклад, матриця Адамара при $n = 4$ має наступний вигляд:

$$A = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix},$$

де коефіцієнт ½ забезпечує властивості ортонормальності і оберненості перетворення.

Матриці A, що має довільні розміри, формуються рекурентною процедурою кронекерівського добутку із матриць меншого розміру. У свою чергу породжується матриця блочної структури [24]. Необхідно відмітити, що матриці Адамара мають додатково ряд інших прикладних властивостей. Вони, доволі часто, мають застосування під час оброблення зображень та дискретних сигналів [25].

Також, у цей самий момент, аналізований опис Z, що представлений у вигляді множини дескрипторів КТ, може бути розглянутий як прямокутна матриця $Z = \left\{z_{i,j}\right\}, i = \overline{1,s},\ j = \overline{1,n}$. Це матриця, що має розміри $s \times n$, рядки якої

містять s дескрипторів опису. Перетворення опису Z тепер може бути подане у вигляді множення квадратної та прямокутної матриць

$$U = Z \times A. \tag{2.5}$$

У результаті множення (2.5) виходить прямокутна матриця $U = \left\{u_{i,j}\right\}, i = \overline{1,s},\ j = \overline{1,n}$. Дана матриця має розміри $s \times n$, а її рядки містять вектори розкладання α для дескрипторів опису Z. Орієнтуючись на той факт, що у даному дослідженні набір ФУ розглядається і застосовується у повному складі, необхідно аналізувати різноманіття способів упорядкування систем ФУ, наприклад, Адамар, Уолш, Пелі або Трахтман. Кожна з ФУ має ряд властивостей, а також відповідні сфери прикладного застосування.

2.2 Класифікація на підставі модифікації простору ознак

Суть традиційного методу класифікації полягає у встановленні ступеня релевантності $\Theta(Z,E_k)$ між описом $\{z_i\}_{i=1}^{s}$ аналізованого зображення та складових компонентів $E_k = \{e_i(k)\}_{i=1}^{s}$ бази $E = \{E_k\}_{k=1}^{N}$ еталонних описів [26]. Клас m об’єкта визначається відповідно до екстремуму функції $\Theta(Z,E_k)$ у вигляді

$$m = \arg \underset{k=1,,N}{\text{extr}}\ \Theta(Z,E_k). \tag{2.6}$$

Після того, як здійснюється трансформація порівнюваних описів, процес класифікації базується на тому, що визначається ступінь релевантності $\Theta^{\alpha}(\alpha(Z),\alpha(E_k))$ для трансформованих описів $\alpha(Z)$ та $\alpha(E_k)$ у новому просторі ознак. У свою чергу, кожен перетворений опис є множиною векторів із компонентами із C^n (у редукованому просторі – C^{α}).

Орієнтуючись на той факт, що, при побудові класифікатора на обмежений діапазон значень вхідного сигналу та фіксоване перетворення у просторі цілих чисел, може існувати можливість оцінки діапазону значень функцій Θ та Θ^{α} і використати ці знання для виконання класифікації.

Метрика для множин векторів є одним із варіантів Θ чи Θ^{α}. Це може бути як і відстань Хаусдорфа, так і міра релевантності для двох множин A,B однотипних векторів

$$X(A,B) = \max\{\max_{a \in A} \varrho(a,B), \max_{b \in B} \varrho(b,A)\}, \tag{2.7}$$

де $\varrho(a,B) = \min_{b \in B} \varrho(a,b)$, а $\varrho(a,b)$ – це метрика, що представлена у векторному просторі. Наприклад, це може бути Манхетенська відстань для C^n, C^{α} або відстань Хемінга для бінарних даних із B^n. Значення міри (2.7) є відстань між двома спеціально обраними точками множин.

Тепер необхідно зробити опис загального вигляду класифікації. На початку, дається простір B^n. Цей простір складається з бінарних багатовимірних векторів, які мають розмірність n. Саме у цьому просторі бінарних векторів будуть створені образи розпізнаваного об’єкту разом з еталонами. На наступному кроці, є необхідність фіксування деякої мультимножини векторів $E_i \subset B^n$, що буде представлено у вигляді деякого опису еталону $E_i = \{e_v(i)\}_{v=1}^{s}$, що знаходиться у просторі множин дескрипторів КТ ($s = card\ E$ – це число дескрипторів у множині [27, 28]). Ознаки це вектори $e_v \in B^n$. У свою чергу, скінченна множина цих векторів створює опис об’єкту. Класифікація, відповідно, передбачає той факт, що є наявність деякої бази E описів еталонних зображень, розміри якої дорівнюють $N : E = \bigcup_{i=1}^{N} E_i$. Варто відмітити, що кожен еталонний опис E_i

має можливість репрезентувати окремий клас, який має вигляд скінченної множини дескрипторів КТ із потужністю S.

Тепер, пропонується розглянути довільний опис $Z \subset B^n$, $Z = \{z_w\}_{w=1}^{s}$ розпізнаваного об'єкту. Необхідно побудувати класифікатор K як відображення $K : Z \to [1,2,...,N]$ на основі попереднього конструювання деякої індексованої структури на множині E. Класифікацію K необхідно представити у вигляді процесу, що складається з двох етапів $K = K_2 K_1$. На першому етапі $K_1 : B^n \to [1,2,...,N]$ здійснюється визначення класу d_w для кожного дескриптора $z_w \in Z$. Після цього, на наступному етапі $K_2 : D \to [1,2,...,N]$ із множини $D = \{d_w\}_{w=1}^{s}$ локальних рішень формується результуюче рішення щодо класу об'єкту Z. Варто відмітити, що K_1 можна розглядати як багатозначну характеристичну функцію, за допомогою якої можна визначити еталонний клас E_i.

Етапи K_1, K_2 будуються за допомогою того, що на етапі K_1 створюється деякий випадковий розподіл за класами еталонів. Також не варто забувати про логічне оброблення таких розподілів [28, 29].

Реалізація K виконується безпосередньо за допомогою апріорних даних. Ці дані належать до існуючої бази E. Це відбувається тому, що належність усіх $e_v(i)$ до відповідного образу E_i всередині бази уже відома на початку класифікації.

У випадку здійснення класифікації K_1 шляхом традиційного лінійного пошуку (метод повного перебору), якщо послідовно переглядавати кожен елемент набору E, тоді може використовуватись конкурентне правило [30]:

$$d_w = \arg\min_{i,\ v} \rho\left(z_w,\ e_v(i) \right), \tag{2.8}$$

де d_w – це номер еталону E_i, до якого буде віднесено дескриптор Z_w об'єкту $d_w \in \{1,...,N\}$;

$\rho(z_w, e_v(i))$ – метрика у векторному просторі.

У свою чергу, необхідну кількість Q обчислених значень метрики в (2.8) лінійним пошуком можна оцінити параметром $Q = N \cdot s^2$. При цьому, обсяги описів еталонів та об'єкта вважаються рівноцінними. Для векторів простору B^n в (2.1) може бути застосована проста, в обчислювальному сенсі, метрика Хемінга. Дана метрика підраховує кількості бітів, які не співпадають.

Далі, на етапі класифікації K_2, усе зводиться до того, що на кожному кроці аналізу опису Z за правилом (2.8) при значенні d_w відбувається інкрементація числа r_i голосів елементів, які віднесені до i-го класу

$$r_i = \left\{ \begin{array}{ll} r_i + 1, & d_w = i \\ r_i, & d_w \end{array} \right\}, \qquad (2.9)$$

а клас i_0 образу Z об'єкта визначається за найбільшої кількості голосів:

$$i_0 = \arg \max_{i=1,...,N} r_i . \qquad (2.10)$$

Значення $\{r_i\}_{i=1}^N$ відображають гістограму класів за числом голосів елементів із Z. Вирази (2.9), (2.10) надають можливість конкретизувати етап K_2. Конкретизація полягає у обробці голосів для компонентів опису Z. Розглянута процедура класифікації базується на принципі інтелектуального аналізу даних. Принцип полягає у підрахунку числа позитивних рішень на аналізованій множині даних [31].

У випадку порівняння двох двійкових або символьних рядків даних має застосування Метрика Хемінга [32]. Під час порівняння двох двійкових

рядків, що мають однакову довжину, відстань Хемінга – це кількість бітових позицій, в яких два біти є відмінними.

$$d_{ij} = \sum_{k=1}^{p} \left| x_{ik} - x_{jk} \right|. \tag{2.11}$$

На рисунку 2.1, у якості прикладу, можна побачити два двійкових ряди, у яких відрізняються 3 пари бітів. Це означає, що у даному випадку відстань Хемінга дорівнює трьом.

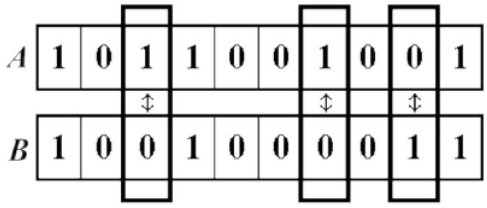


Рисунок 2.1 – Два бітових ряди, чия відстань Хемінга дорівнює 3

У випадку розглядання відстані Хемінга у більш загальному випадку, то дана метрика має застосування для рядків однакової довжини будь-яких *q*-ічних алфавітів. Вона служить у якості метрики відмінності (функцією, яка може визначати відстань в метричному просторі) об’єктів однакової розмірності. Далі наведено приклад відстаней в двійковому тессеракті на рисунку 2.2.

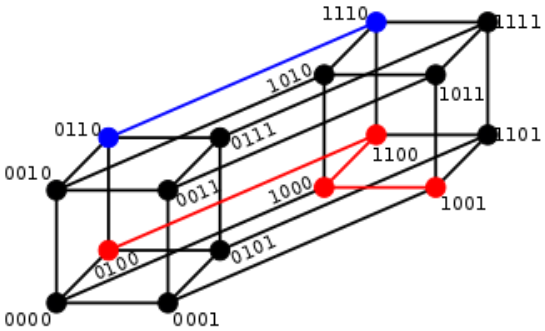


Рисунок 2.2 – Приклади відстаней в двійковому тессеракті: відстань 1 (синя) 0110 → 1110, відстань 3(червона) 0100 → 1001

Манхетенська відстань обчислює відстань між двома векторами, що складаються з цілих або дійсних чисел. Ця відстань обчислюється як сума абсолютних різниць між двома векторами.

$$d_1(p,q) = \left|\left|p-q\right|\right|_1 = \sum_{i=1}^{n} \left|p_i - q_i\right|, \tag{2.12}$$

де (p,q) – це вектори $p=\left(p_1, p_2, \ldots, p_n\right)$ та $q=\left(q_1, q_2, \ldots, q_n\right)$.

На рисунку 2.3 наведено два ряди цілих чисел, у яких відрізняються 3 пари чисел. Абсолютна різниця між цими двома векторами, у випадку використання Манхетенської метрики, складає 14.

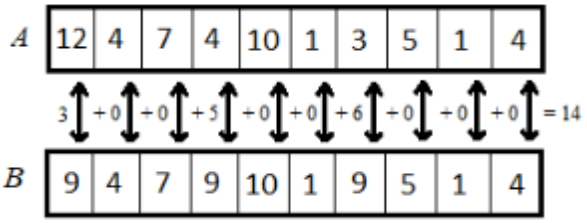


Рисунок 2.3 – Два цілочисельних ряди з Манхетенською відстанню 14

Іншим варіантом оцінювання релевантності є відстань Танімото (Жаккара) для множин, яка означає відношення числа елементів симетричної різниці та об’єднання множин

$$T(A,B) = \frac{card(A \Delta B)}{card(A \cup B)}. \tag{2.13}$$

Міра (2.13) на відміну від (2.7) відображає кількісні характеристики еквівалентних та відмінних елементів порівнюваних множин.

У випадку використання метрики (2.13) важливим моментом є її граничне значення, для якого елементи векторного простору вважаються еквівалентними. У багатовимірних просторах даних – це одна з ключових

проблем. У свою чергу, на результат класифікації вибір порогу має суттєвий вплив.

Для того, щоб вирішити дану проблему пропонуються два способи. Перший спосіб визначає граничне значення

$$\varrho_{\lim}(a,b) = \alpha\varrho_{max}(a,b) \tag{2.14}$$

як відсоток α від максимально теоретично визначеного значення метрики. Наприклад, для метрики Хемінга, що у даному випадку змінюється на інтервалі [0, 256], при відсотку у α=25% можна визначити $\varrho_{\lim}(a,b) = 0{,}25 \times 256 = 64$.

Більш практичним є спосіб, що визначає $\varrho_{\lim}(a,b)$ за результатом аналізу конкретних експериментальних даних, де ϱ_{max} обчислюється для фіксованого набору описів еталонів. Час обчислення ϱ_{max} ніяк не впливає на витрати для класифікації та здійснюється на попередньому етапі аналізу даних.

Пропонується оцінювати результативність методу класифікації показником точності pr. Він обчислюється у вигляді відношенням числа правильно класифікованих об’єктів r_p до їх загального числа r, що використовувалося в експерименті

$$pr = r_p / r. \tag{2.15}$$

Важливо пам’ятати, що завадостійкість також є важливим показником результативності методів розпізнавання зображень. Цей показник характеризується значенням точності класифікації в умовах дії завад [33]. У тому випадку, якщо дія адитивної завади на зображення $B(x,y)$ описується моделлю, що має вигляд

$$B_{\xi}(x,y) = B(x,y) + \xi(x,y), \tag{2.16}$$

а завада $\xi(x,y)$, у свою чергу, характеризується середньоквадратичним відхиленням σ, тоді відношення сигнал-шум може бути описано у вигляді

$$\mu = B_m / \sigma,$$

де B_m – це амплітудна характеристика (наприклад, середнє значення яскравості). Важливим моментом є той факт, що вивчення залежності $pr(\mu)$ для розробленого методу.

2.3 Апарат функцій Уолша

Уолш [34] отримав повну систему ортонормованих прямокутних функцій, яка доповнює систему функцій Радемахера і тепер відома як система функцій Уолша. Множину функцій Уолша зазвичай розділяють на три групи, які відрізняються порядком положення окремих функцій у системі. Загальноприйняті наступні упорядкування: 1) упорядкування за частотою (за Уолшем); 2) діадичне упорядкування (за Пелі); 3) природне упорядкування (за Адамаром).

Упорядкування за частотою або за Уолшем. Це упорядкування було запропоноване Уолшем [34]. Будемо позначати множину функцій Уолша, що упорядковані, таким чином через

$$S_{\omega} = \left\{ wal_{\omega}(i,t),\ i = 0,\ 1,\ \ldots,\ N-1 \right\}, \tag{2.17}$$

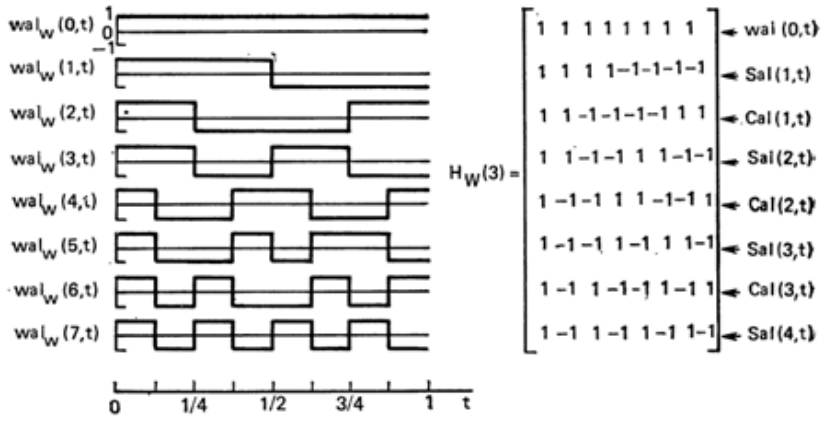
де $N = 2^n$, $n = 1, 2, 3, \ldots$; нижній індекс ω означає упорядкованість за Уолшем, а i відповідає i-му елементу S_{ω}. Якщо через s_i позначити частоту $wal_{\omega}(i,t)$, тоді S_i визначається як

$$S_i = \begin{cases} 0, i = 0; \\ i/2, i - \text{парне}; \\ (i+1)/2, i - \text{непарне}. \end{cases} \quad (2.18)$$

Функція cal та sal, які відповідають $wal_\omega(i,t)$, описуються наступним чином:

$$\begin{aligned} cal(S_i, t) &= wal_\omega(i,t), i - \text{парне}; \\ cal(S_i, t) &= wal_\omega(i,t), i - \text{непарне}. \end{aligned} \quad (2.19)$$

Перші вісім функцій Уолша наведені на рисунку 2.4 (а), на якому видно, що частота наступної функції Уолша більша або дорівнює частоті попередньої функції Уолша та має точно на один перетин нульового рівня більше у відкритому інтервалі $t \in (0, 1)$. Звідси виходить назва упорядкованості за частотою. Елемент S_ω можна отримати з множини функцій Радемахера при використанні коду Грея [35].



а) б)

Рисунок 2.4 – Функції Уолша, що упорядковані за Уолшем, при *N*=8:
а) безперервні; б) дискретні

Дискретизація функції Уолша, що зображені на рисунку 2.4 (а), у восьми рівновіддалених точках призводить до матриці (8×8), що показана на

рисунку 2.4 (б). У загальному випадку виходить матриця $(N \times N)$. Такі матриці позначають $H_{\omega}(n), n = \log_2(N)$, оскільки вони виходять у результаті переупорядкування рядків матриці Адамара.

Діадичне упорядкування було введено Пелі [36]. Функуції Уолша є елементами діадичної групи та можуть бути упорядковані за допомогою коду Грея [37]. Ця множина функцій Уолша позначається як

$$S_p = \left\{ wal_p(i, t), i = 0, 1, \ldots, N-1 \right\}, \tag{2.20}$$

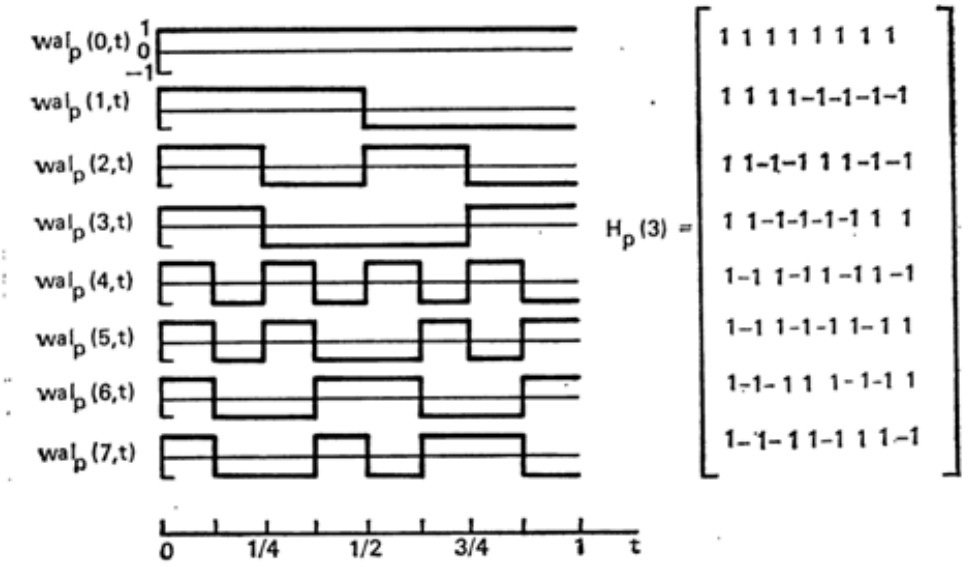
де індекс *p* упорядкування за Пелі, а *i* означає *i*-ий елемент S_p. Множина S_p пов'язана з множиною S_w, яке упорядковане за Уолшем, відношенням

$$wal_p(i, t) = wal_w[b(i), t], \tag{2.21}$$

де $b(i)$ – перехід від коду Грея до бінарного коду з індексом *i*. Проілюструємо відношення (2.21) при параметрі $N = 8$. Відповідні результати наведені на рисунку 2.5. Використовуючи дані, що наведені на рисунку 2.5, до функцій $wal_w(i, t), i = 0, 1, \ldots, 7$, що були показані на рисунку 2.4 (а), отримуємо вісім функцій Уолша $wal_p(i, t)$, що наведені на рисунку 2.6 (а).

i_{10}	i_2	$b(i)_2$	$b(i)_{10}$	Формула (2.21)
0	000	000	0	$\text{wal}_p(0,t) = \text{wal}_w(0,t)$
1	001	001	1	$\text{wal}_p(1,t) = \text{wal}_w(1,t)$
2	010	011	3	$\text{wal}_p(2,t) = \text{wal}_w(3,t)$
3	011	010	2	$\text{wal}_p(3,t) = \text{wal}_w(2,t)$
4	100	111	7	$\text{wal}_p(4,t) = \text{wal}_w(7,t)$
5	101	110	6	$\text{wal}_p(5,t) = \text{wal}_w(6,t)$
6	110	100	4	$\text{wal}_p(6,t) = \text{wal}_w(4,t)$
7	111	101	5	$\text{wal}_p(7,t) = \text{wal}_w(5,t)$

Рисунок 2.5 – Відношення між функціями Уолша, що упорядковані за Уолшем та упорядковані за Пелі



а) б)

Рисунок 2.6 – Функції Уолша, що упорядковані за Пелі при N = 8:
а) безперервні; б) дискретні

У дискретному випадку, виконуючи дискретизацію функцій Уолша (рисунок 2.6 (а)), отримаємо матрицю (8×8), яка наведена на рисунку 2.6 (б). Цю матрицю також можна отримати переупорядковуючи рядки матриці Адамара (8×8). Матриці, що пов’язані з функціями Уолша, які упорядковані за Пелі, будемо позначати $H_p(n), n=\log_2(N)$. Елементи $h_{uv}^{(p)}$ матриці $H_p(n)$ можна отримати з наступної формули:

$$h_{uv}^{(p)} = (-1)^{\sum_{i=0}^{n-1} u_{n-1-i} v_i}, \; u, v = 0, 1, \ldots, N-1. \qquad (2.22)$$

Природнє упорядкування або упорядкування за Адамаром. Множина функцій Уолша позначається наступним чином:

$$S_h = \left\{ wal_h(i, t), i = 0, 1, \ldots, N-1 \right\}, \qquad (2.23)$$

де індекс *h* означає упорядкування за Адамаром, *i* означає *i*-ий елемент S_h. Функції, що належать S_h, пов'язані з функціями, що упорядковані за Уолшем, відношенням

$$wal_h(i,t) = wal_w[b(<i>)t], \tag{2.24}$$

де $<i>$ – бінарно-інвертований запис *i*, а $b(<i>)$ – перехід від коду Грея до бінарного коду $<i>$. Проілюструємо цей перехід, який описаний відношенням (2.24) для $N = 8$; результати розрахунку наведені на рисунку 2.7.

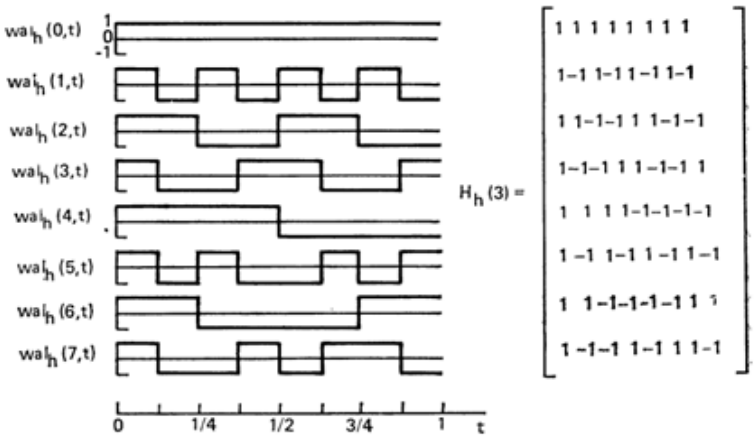
i_{10}	i_2	$b(i)_2$	$b(i)_{10}$	Формула (2.24)
0	000	000	0	$wal_p(0,t) = wal_w(0,t)$
1	001	001	1	$wal_p(1,t) = wal_w(1,t)$
2	010	011	3	$wal_p(2,t) = wal_w(3,t)$
3	011	010	2	$wal_p(3,t) = wal_w(2,t)$
4	100	111	7	$wal_p(4,t) = wal_w(7,t)$
5	101	110	6	$wal_p(5,t) = wal_w(6,t)$
6	110	100	4	$wal_p(6,t) = wal_w(4,t)$
7	111	101	5	$wal_p(7,t) = wal_w(5,t)$

Рисунок 2.7 – Відношення між функціями Уолша, що упорядковані за Уолшем та упорядковані за Адамаром

Використовуючи дані рисунку 2.7 та функції $wal_w(i,t), i = 0, 1, \ldots, 7$, можна отримати перші вісім функцій Уолша (рисунок 2.8 (а)).

У дискретному випадку дискретизація функції Уолша призводить до матриці Адамара (8×8), що зображена на рисунку 2.8 (б). У загальному вигляді виходить матриця $H_h(n)$ розміром $(N \times N)$, де $n = \log_2 N$. Для цього класу матриць Адамара буде справедливим розбиття на підматриці, що мають вигляд

$$H_h(n) = \begin{bmatrix} H_h(n-1) & H_h(n-1) \\ H_h(n-1) & -H_h(n-1) \end{bmatrix}. \tag{2.25}$$



а) б)

Рисунок 2.8 – Функції Уолша, що упорядковані за Адамаром при $N = 8$: а) безперервні; б) дискретні

Такі матриці відповідають природньому упорядкуванню.

2.4 Редукція простору ознак

Визначення трансформованої системи ознак засновано на інтеграційному процесі розкладання даних вхідного опису за спектром ортогональних проекцій. Зважаючи на цей факт, можна судити, що у новому просторі C^n з'являється можливість формування компактного подання $\alpha \times (Z)$ та $\alpha \times (E_k)$ зі стисненим об'ємом коефіцієнтів.

У ході зменшення розміру вектору ознак, стиснення розмірності простору даних дає можливість реалізувати перспективну ідею скорочення обсягу обчислень для значення релевантності Θ^{α}, а також заради зменшення часу, необхідного для виконання класифікації загалом [38]. Дане оброблення набуло популярності у спектральних методах. Саме там, шляхом фільтрації незначущих компонентів спектру, за рахунок деякого збільшення помилки відновлення сигналу, зменшується надмірність представлення та скорочується вектор оброблюваних даних. Варто відмітити, що при цьому, показники функціонування систем знижуються незначним чином.

Варіантом одного з традиційних способів є ідея, яка включає відбір серед множини $\{\alpha_i\}_{i=1}^{s}$ тієї множини, яка буде найбільш інформативною до її підмножини $\{\alpha_i\}_{i=1}^{q}$, $q \ll s$. Також, цей спосіб може забезпечити достатньо якісну класифікацію. При цьому, формування стисненої підмножини $\{\alpha_i\}_{i=1}^{q}$ дає можливість відбору компактної частини серед функцій ортогональної системи W. Мається на увазі можливість використання для класифікації лише деякої підмножину W_q із W, $W_q \subset W$. У результаті, зникає необхідність формування повної системи коефіцієнтів розкладання як для вхідного образу, так і для еталонної множини E.

У свою чергу, те ж саме оброблення можна було б спробувати здійснити і безпосередньо для вхідної множини $\{z_i\}_{i=1}^{s}$ ознак. Наприклад, за встановленим критерієм інформативності. Однак, варто відмітити, що таке безпосереднє скорочення, доволі часто, прямо пропорційно до кількісного скорочення опису, призводить до суттєвих втрат інформації. Одночасно з цим, обчислені проекції $\{\alpha_i\}_{i=1}^{s}$ не тільки мають багатоспектральний сенс даних, але також містять, у більшій мірі, інтегровану інформацію за рахунок процесу розкладення (2.1) за системою базисних функцій. Саме тому вони можуть бути успішно покладені в основу здійснення ефективного стиснення даних.

Таким чином, якщо введеною процедурою фільтрації вдасться відібрати високоінформативну частину спектру, яка забезпечує достатньо якісну класифікацію, тоді, за рахунок цього, можна суттєво скоротити обчислювальні затрати на класифікацію із забезпеченням необхідного рівня результативності.

У ролі результату навчання на заданій множині еталонних сигналів буде виступати отримана у ході такого секвестрування підмножина найбільш інформативної частини спектру ФУ. У свою чергу, її формування напряму залежить від аналізованого простору даних і адаптоване до нього. Для інших

варіантів колекцій еталонних даних підмножина компонентів спектру може мати зовсім інший склад.

Поставимо задачу шляхом трансформації у новий простір на основі побудови відображення $T:\alpha(Z)\to\alpha\times(Z)$, або фактично $T:W\to W_q$, за рахунок застосування ортогонального розкладання за компактною системою W_q ФУ сформувати стиснені інформативні описи $\alpha\times(Z)$ та $\alpha\times(E_k)$. Саме вони забезпечують достатні показники результативності класифікації. Дане відображення $T=T(E,W)$ є функцією як еталонних даних E, так і виду ортогональної системи функцій W.

Внаслідок наявності значних кореляційних зв'язків між елементами зображення реальної природи, що знаходить своє відтворення у значеннях дескрипторів КТ, основна енергія у дискретному спектрі має тенденцію концентруватися у відносно невеликій кількості відліків, що відповідають повільно осцилюючим базовим функціям. Саме тому, без істотної шкоди для відновлення чи класифікації зображення, невеликі за величиною спектральні коефіцієнти, можна взагалі обнулити (або відкинути їх аналіз). У свою чергу, значущі елементи спектру оцифрувати та використовувати у процесі розпізнавання.

На сьогоднішній день, основним апаратом для оцінювання значущості обчислених коефіцієнтів ортогонального перетворення у моделі подання сигналу є дисперсійний аналіз цих коефіцієнтів (2.1), оскільки середньоквадратична помилка відновлення напряму напряму залежить від дисперсійних характеристик. Саме тому, найбільше значення дисперсії є природним критерієм для вибору множини значимих коефіцієнтів розкладення. Зазвичай, обчислюють криву нормованих значень дисперсій в залежності від номера ФУ, розміщених за спаданням, і впроваджують у процес класифікації підмножину коефіцієнтів з найбільшими значеннями.

У ході застосування даного апарату в рамках задачі класифікації, де описи зображень подаються у вигляді множини дескрипторів, мають місце свої можливості та особливості. Справа полягає в тому, що коефіцієнти з

найбільшими дисперсіями не завжди можуть забезпечити потрібний рівень розрізнення зображень, що подаються на вході. Іншим фактором є те, що є можливість здійснення дисперсійного аналізу даних як у рамках повної бази еталонів, так і в межах окремих її представників. А це, у свою чергу, може покращити показники результативності. Також, ще однією можливістю є організація дисперсійного аналізу в рамках системи фрагментів дескрипторного опису, що реалізує віконне оброблення [39].

На повній множині даних E для описів еталонів отримаємо матрицю (2.5), що має розмір $Ns \times n$. Тепер необхідно обчислити для кожного стовпця матриці U вектори математичного очікування Mu_i та дисперсії σ_i^2

$$Mu_i = \frac{1}{Ns}\sum_{j=1}^{Ns} u_{i,j}, \quad \sigma_i^2 = \frac{1}{Ns-1}\sum_{j=1}^{Ns}(u_{i,j} - Mu_i)^2. \qquad (2.26)$$

Під час визначення показників (2.26) для окремого еталону, необхідно взяти $N = 1$. На основі отриманих дисперсійних характеристик можна побудувати класифікатор. Він є аналогічним класифікатору за значенням дисперсій безпосередньо для множини дескрипторів опису Z [40-43].

На наступному кроці необхідно здійснити ранжирування списку ФУ w_i за значенням дисперсії σ_i^2 для окремих компонент спектру: $w_1, w_2, , w_n$, так щоб при цьому виконувалась умова $\sigma_1^2 \geq \sigma_2^2 \geq \ldots \geq \sigma_n^2$ спадання значень. Тепер, отриманий список можна покласти в основу процедури стиснення.

Варто відмітити, що є інший варіант для попереднього аналізу. Це процес обчислення значення дисперсії для множини квадратів коефіцієнтів розкладання. У такій ситуації, у виразах (2.13) необхідно застосувати значення $u_{i,j}^2$ замість $u_{i,j}$ [44].

3 АНАЛІЗ РЕЗУЛЬТАТІВ ЕКСПЕРИМЕНТАЛЬНОГО ДОСЛІДЖЕННЯ МЕТОДУ

3.1 Аналіз вибору програмного середовища

У рамках кваліфікаційної роботи був розроблений алгоритм для класифікації зображень шляхом впровадження апарату розкладання компонентів опису з системою ортогональних функцій Уолша та застосування моделей стиснення простору ознак. Для реалізації роботи було обрано бібліотеку OpenCV. Даний варіант має відкритий доступ, а тому є можливість безкоштовно використати цю бібліотеку та вільно продавати програмне забезпечення, яке було написане з її використанням. Варто відмітити високу швидкість, що обумовлено високопродуктивною мовою C++, й, на кінець, бібліотека містить великий обсяг готових рішень, які пов’язані з обробленням зображень [45].

OpenCV (Open Source Computer Vision Library) – це бібліотека програмного забезпечення з відкритим кодом для комп’ютера та машинного навчання. Бібліотека містить модулі, які охоплюють різноманітні області комп’ютерного зору. Постійно розвивається підтримка інтерфейсу із мовою програмування Python, яка є більш простою та зручнішою у використанні. Цей інтерфейс продовжує розроблюватись – ще не всі частини OpenCV охоплені. Але у майбутньому положення, обов’язково, зміниться, тому що інтерфейс активно розвивається.

Далі, у якості прикладу на рисунку 3.1 наведено зображення, на якому було побудовано 500 дескрипторів за допомогою бібліотеки OpenCV.

Рисунок 3.1 – Результат побудови ключових точок
за допомогою бібліотеки OpenCV

Причини вибору мови програмування Python:

– мова Python є синтаксично простою та зручною для програмної реалізації математичних задач. Завдяки простоті навчання та використання, скрипти, що написані мовою Python, можна легко писати та виконувати набагато швидше, ніж інші мови програмування. Передбачуване виконання програм є важливою перевагою для побудови систем реального часу;

– мова Python є надзвичайно універсальною. Вона може використовуватися в усіх можливих різновидах середовищ, таких як мобільні застосунки, настільні програми, веб-розробка, апаратне програмування та багато іншого. Універсальність Python робить її більш привабливою для використання завдяки великій кількості доповнень;

– Python була створена більше 30 років тому – це доволі великий проміжок часу для того, щоб будь-яке співтовариство мови програмування адекватно зросло та дозріло для підтримки розробників, починаючи від початківців і закінчуючи рівнями експертів. Вагомою перевагою є той факт, що була створена велика кількість документації, посібників та відео-посібників для мови Python, за допомогою яких навчаються та створюють продукти. Це означає, що кожен розробник будь-якого рівня кваліфікації та

віку може використовувати та отримувати підтримку, необхідну для того, щоб підвищити свої знання з мови програмування [46];

– хмарні обчислення, машинне навчання та великі дані – на сьогоднішній день, це одні з найгарячіших тенденцій у світі комп'ютерних наук. Це допомагає багатьом організаціям трансформувати та вдосконалювати свої внутрішні та робочі процеси. Сотні бібліотек Python щодня використовуються у тисячах проектів машинного навчання, таких як TensorFlow для нейронних мереж та OpenCV для комп'ютерного зору тощо;

– завдяки великій підтримці спільноти та корпоративній спонсорській підтримці, дана мова програмування має чудові бібліотеки. Ці бібліотеки допомагають значно заощадити час та зусилля. Варто також відмітити велику кількість хмарних медіа-сервісів, які пропонують підтримку між платформами за допомогою бібліотечних інструментів;

– мова Python також допомагає в області автоматизації завдань. Це обумовлено існування великої кількості відповідних інструментів та модулів. Завжди у пріоритеті ті засоби, за допомогою яких можна легко досягти просунутого рівня автоматизації завдяки простому використанню необхідних скриптів, що написані відповідною мовою програмування. Python – це один з найкращих прискорювачів продуктивності в автоматизації тестування програмного забезпечення. Вражаючим фактом є те, наскільки менше часу та кількох рядків потрібно для написання кодів засобів автоматизації у процесі використання Python.

3.2 Особливості моделювання і експериментального дослідження

Для роботи програмного забезпечення та алгоритмів створено базу вхідних зображень. Ця база складається з п’яти еталонів. Кожен з еталонів – це зображення, герби футбольних клубів, що мають розміри 325×325 пікселів. Окремий візуальний об’єкт (еталон) розташовується на світлому фоні (рис. 3.2).

а) б) в)

г) ґ) д)

Рисунок 3.2 – Еталонні зображення:
а), б), в), г), ґ) еталони; д) координати ORB-дескрипторів

Також для виконання класифікації на вхід подаються зашумовані зображення з різними коефіцієнтами зашумованості σ. Ці зображення були спеціально модифіковані за допомогою адитивного шуму, щоб перевірити та порівняти якість та час виконання класифікації зображень за допомогою

традиційних методів класифікації зображень та методів із використанням систем ортогональних функцій для формування простору ознак.

Далі наведено скрипт, який виконує зашумовування зображення та результати зашумовувування (рис. 3.3) з різними коефіцієнтами.

Лістинг 3.1 Скрипт для додавання шуму до зображення:

```
def add_noise(img, mean, sigma, X, Y):
    gaussian = np.random.normal(mean, sigma, (X, Y))  # np.zeros((224, 224),
np.float32)

    noisy_image = np.zeros(img.shape, np.float32)

    if len(img.shape) == 2:
        noisy_image = img + gaussian
    else:
        noisy_image[:, :, 0] = img[:, :, 0] + gaussian
        noisy_image[:, :, 1] = img[:, :, 1] + gaussian
        noisy_image[:, :, 2] = img[:, :, 2] + gaussian

    cv2.normalize(noisy_image, noisy_image, 0, 255, cv2.NORM_MINMAX, dtype=-1)
    noisy_image = noisy_image.astype(np.uint8)

    return noisy_image
```

а) б)

в) г)

Рисунок 3.3 – Зашумовані зображення з різними коефіцієнтами σ:

а) σ = 5; б) σ = 20; в) σ = 50; г) σ = 100

За допомогою бібліотеки OpenCV визначаються дескриптори КТ для всіх зображень – як для звичайних, так й для зашумованих. На кожному з них будуються 500 ключових точок – ця цифра чітко фіксована кількість дескрипторів. Це означає, що кількість дескрипторів, які належать будь-якому вхідному зображенню, дорівнює 500 ключових точок.

Тестові зображення, на яких демонструється результат роботи знаходження особливих точок за допомогою бібліотеки OpenCV, наведені у Додатку А (рис. А.2).

Для побудови дескрипторів ключових точок використовується бібліотека мови програмування Python, чий метод повертає колекцію дескрипторів, кожен з яких складається з тридцяти двох десяткових чисел. Для подальших розрахунків та досліджень, у першу чергу, необхідно конвертувати значення дескрипторів у бітовий вигляд. Для цього було створено наступну програмну реалізацію

Лістинг 3.2 Скрипт для конвертування числа у бітовий вигляд:

```
def bitfield(n):
    result = [int(digit) for digit in bin(n)[2:]]
    len_result = len(result)
    new_result = []
    if len_result < 8:
        amount_of_zeros = 8 - len_result
        for i in range(amount_of_zeros):
            new_result.append(0)
        for i in range(len(result)):
            new_result.append(result[i])
        return new_result
    return result
```

Результат роботи методу bitfield(), який конвертує десятичне число у бітовий вигляд наведено на рисунку 3.4.

```
Enter a number to convert it -> 7
[0, 0, 0, 0, 0, 1, 1, 1]

Enter a number to convert it -> 180
[1, 0, 1, 1, 0, 1, 0, 0]
```

Рисунок 3.4 – Результат роботи методу bitfield, який конвертує десятичне число у бітовий вигляд

Лістинг 3.3 Скрипт для перетворення дескриптора у бітовий вигляд:

```
def return_array_of_256_bits(point):
    result_array = []
    for byte in range(len(point)):
        bit_point = bitfield(point[byte])
        for x in range(len(bit_point)):
            result_array.append(bit_point[x])
    return result_array
```

Лістинг 3.4 Скрипт для перетворення усіх дескрипторів еталону у бітовий вигляд:

```
def convert_32_descriptors_to_256_bit(descriptors):
    result_matrix = []
    for keypoint in range(len(descriptors)):
        a = return_array_of_256_bits(descriptors[keypoint])
        result_matrix.append(a)
    return result_matrix
```

У результаті виконання скриптів, що наведено вище, маємо масив дескрипторів зображення, який представлено у бітовому вигляді.

Також, для розрахунків, необхідно мати масив дескрипторів, що складаються з цілісних чисел. Для цього достатньо здійснити перетворення Уолша із використанням матриці Адамара – це квадратична матриця розмірами $n \times n$, що складається із чисел 1 та –1, стовпці яких є ортогональними. Суть перетворення полягає в тому, що кожен дескриптор, який представлений у вигляді бітового вектору, що складається з 256 елементів, множиться на матрицю Адамара розмірами 256×256.

Для цього було реалізовано наступний скрипт.

Лістинг 3.5 Перетворення Уолша за допомогою матриці Адамара:

```
def convert_bit_format_descriptors_to_integers_format(descriptors_bit_format):

    adamar_matrix = hadamard(256)
    converted_descriptors = []
    for i in range(len(descriptors_bit_format)):
        # print(i)

converted_descriptors.append(converted_descriptor_by_adamar_matrix(descriptors
_bit_format[i], adamar_matrix))
    return converted_descriptors
```

```
def convert_descriptor_by_adamar_matrix(descriptor, adamar_matrix):
    result_array = []
    for i in range(len(descriptor)):
        sum = 0
        for j in range(len(descriptor)):
            sum = sum + (descriptor[j] * adamar_matrix[i][j])
        result_array.append(sum)
    return result_array
```

Результати перетворення одного з дескрипторів у бітове (рис. 3.5) та цілісне (рис. 3.6) представлення наведено нижче.

0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1,
0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1,
0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0,
0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0,
0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1,
0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1,
0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0,
1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1,
1, 1, 0, 0, 1, 0

Рисунок 3.5 – Бітове представлення дескриптора

132, 4, 14, -2, -8, -4, 6, -6, 0, 8, 2, 10, -4, -16, 2, 6, -8, -8, 14, -2, 0, 4, 2, -10,
-8, 8, -18, -2, 8, 4, -6, 6, 2, 2, 4, 12, 6, 10, -4, -8, 2, 10, -4, 12, -10, 10, 4, 0, 2,
10, -8, -8, 18, -18, 4, 8, 14, -10, 4, 4, -2, 2, 0, -4, 8, 12, 2, -10, 0, 8, 6, -2, 0, -
12, 2, 6, -8, -8, -10, -10, -4, -8, 2, -2, 0, -16, -6, 10, -8, -4, -2, 2, -4, -4, -10, 6,
-10, -6, -8, -12, 6, -2, 4, 4, 2, -10, -4, -8, -6, 10, 0, -8, 6, 10, 12, 8, -6, 18, 4, 4,
-2, -6, -12, -8, 10, 2, 4, 4, -2, 22, 4, 4, -6, -2, -4, 8, 2, -6, -24, -8, -2, -6, 0, 4,
18, -6, 4, -12, 2, 6, -8, 4, -14, 2, -4, 4, 2, 6, 0, 12, 0, 8, -2, -26, -4, 0, -10, 10,
0, -8, -2, -10, 4, -16, -2, 10, 0, 0, 2, -14, -8, -12, -2, -6, -12, -4, 14, 6, 4, 0, 18,
-2, -2, -6, -4, 0, 6, -2, -8, -8, -2, -14, -4, -8, -2, 6, 0, 0, 10, -18, 4, 0, -2, -2, -12,
12, 6, -6, 8, -12, 18, -6, -16, 0, 8, 4, -18, -6, 8, 0, 2, -6, 4, 8, -6, -2, 4, -4, -2, -
10, 16, -4, -6, -10, 4, -4, 10, 2, 0, -4, 2, -2, 4, 4, 2, 2

Рисунок 3.6 – Цілісне представлення дескриптора

Тепер, маючи усі необхідні дані, можна на практиці порівняти класичні методи класифікації зображень та методи із використанням систем ортогональних функцій для формування простору ознак. Дослідження

полягає у вимірюванні швидкості виконання та точності класифікації за результатами моделювання кожного з методів.

Маючи цілочисельне представлення дескрипторів після перетворень Уолша, можна порахувати список ФУ за величиною квадрату дисперсії на множині дескрипторів бази еталонів. Важливо відсортувати даний список за зменшенням та відібрати перші 16 ФУ. Далі наведено скрипт, який виконує розрахування та сортування списку ФУ за величиною квадрату дисперсії на множині дескрипторів бази еталонів.

Лістинг 3.6 Реалізація розрахунку дисперсії та сортування ФУ:

```
def get_dispersion_for_etalon(etalons_descriptors, index, degree):

    sum_for_average = 0
    average = 0

    # the average value of the column is calculated.
    for j in range(len(etalons_descriptors)): # range 1000
        sum_for_average = sum_for_average + etalons_descriptors[j][index]
    average = sum_for_average / len(etalons_descriptors)

    sum = 0

    #   The average value is subtracted from each element of the column.
    #   The difference is squared. Each difference for each element is summed.
    if degree:
        for j in range(len(etalons_descriptors)):
           sum = sum + (((etalons_descriptors[j][index] ** 2) - (average **
2)) ** 2)
    else:
        for j in range(len(etalons_descriptors)):
             sum = sum + ((etalons_descriptors[j][index] - average) ** 2)

    res = sum / (len(etalons_descriptors) - 1)
    return round(res, 2)


def get_list_of_dispersions(etalons_descriptors, degree: bool):
    list_of_dispersions = []
        for i in range(len(etalons_descriptors[0])):

list_of_dispersions.append(get_dispersion_for_etalon(etalons_descriptors, i,
degree))

    return list_of_dispersions


top_sorted_dictionary_of_dispersions_divided_by_max = \
dict(itertools.islice(sorted_dictionary_of_dispersions_divided_by_max.items(),
1,17))
```

У результаті маємо таблицю 3.1, у якій знаходяться номери перших 16-ти ФУ, що були відсортовані за величиною квадрату дисперсії на множині дескрипторів бази еталонів.

Таблиця 3.1 – Номери відсортованих ФУ

№	№ ФУ	Нормована дисперсія
1	0	1,0
2	45	0,00159
3	106	0,00157
4	208	0,00156
5	176	0,00145
6	10	0,00134
7	85	0,00128
8	77	0,00123
9	109	0,00122
10	80	0,00119
11	209	0,00119
12	9	0,00115
13	3	0,00113
14	54	0,00107
15	72	0,00107
16	197	0,00105

Для подальшої класифікації потрібні індекси ФУ, що наведені у таблиці 3.1. Ідея полягає в тому, щоб використовувати елементи, що мають такі самі індекси як у таблиці 3.1, з вектору цілочисельних або бітових представлень дескриптора при виконанні порівняння двох дескрипторів між собою. Тобто, пропонується, щоб при порівнянні двох дескрипторів, замість порівняння усіх 256 елементів масиву, порівнювались лише 16. Теоретично, це може дозволити скороти час виконання класифікації у декілька разів, при цьому, значно не втрачаючи показники точності.

Альтернативно розглянуто два варіанти порівняння еталонних описів між собою. Перший варіант – використання відстані Хаусдорфа, а другий варіант – використання відстані Танімото.

3.3 Аналіз результатів використання відстані Хаусдорфа

Спробуємо для початку використати відстань Хаусдорфа. Почнемо з класичної метрики де порівнюються дескриптори, що зберігаються у бітовому виді. При використанні даного методу знаходження відстані між дескрипторами використовується метрика Хемінга. Далі наведено скрипт даної метрики.

Лістинг 3.7 Реалізація метрики Хемінга:

```
def my_hamming_distance(first_value, second_value):
    return float(len(first_value)) * distance.hamming(first_value,
second_value)
```

Також була реалізована відстань Хаусдорфа, яка має наступний вигляд.

Лістинг 3.8 Реалізація відстані Хаусдорфа:

```
def get_minimum_distance_for_descriptor_among_many(descriptor, many,
distance_type):
    if distance_type == DistanceType.MANHATTAN:
        min_distance = 100000
        for i in range(len(many)):
            distance_between_descriptors = manhattan_distance(descriptor,
many[i])
            if distance_between_descriptors < min_distance:
                min_distance = distance_between_descriptors
        return min_distance
    if distance_type == DistanceType.HAMMING:
        min_distance = 256
        for i in range(len(many)):
            distance_between_descriptors = my_hamming_distance(descriptor,
many[i])
            if distance_between_descriptors < min_distance:
                min_distance = distance_between_descriptors
        return min_distance
def get_minimum_distance_for_each_many_descriptors_among_many(many_A, many_B,
distance_type):
    array_of_min_distances = []
    for i in range(len(many_A)):
        array_of_min_distances.append(
            get_minimum_distance_for_descriptor_among_many(many_A[i], many_B,
distance_type))
    return array_of_min_distances
def get_maximum_between_two_arrays_of_minimums(array_A, array_B, nameA,
nameB):
    max_A = max(array_A)
    max_B = max(array_B)
    return max_A if max_A > max_B else max_B
```

Варто відмітити, що реалізація відстані Хаусдорфа є універсальною і дозволяє користуватись метриками Хемінга або Манхетенською на вибір.

У результаті проведення експериментів, було встановлено, що відстань Хаусдорфа між еталонними описами з дескрипторами, представлених у бітовому виді знаходиться у вузькому інтервалі значень [95,…, 100]. При цьому, комп'ютерний час для обчислення метрики склав 21 секунду. На рисунку 3.7, що наведено нижче, у якості прикладу, можна побачити відстані Хаусдорфа, що були отримані у результаті порівняння звичайного еталонного опису та зашумованих з коефіцієнтом зашумованості σ = 10. Наприклад, вираз, що має вигляд «A – A : 88,0» означає, що була знайдена відстань Хаусдорфа між еталонним описом звичайного зображення A та еталонним описом зашумованого зображення A, а сама відстань між описами дорівнює 88.

```
{'A - A': 88.0, 'A - B': 98.0, 'A - C': 96.0, 'A - D': 98.0, 'A - E': 95.0}
```

Рисунок 3.7 – Результат порівняння еталонного опису із зашумованими описами σ = 10 за допомогою відстані Хаусдорфа

Як можна побачити на рисунку 3.7, у ході порівняння еталонних описів, відстань між звичайним еталонним описом A та зашумованим описом A – є найменшою, а це означає, що дані описи є найбільш схожими.

Далі, на рисунку 3.8, у якості прикладу наведено загальну матрицю порівнянь зашумованих та звичайних еталонних описів, яка було отримана у ході використання відстані Хаусдорфа між еталонними описами з дескрипторами у бітовому виді при коефіцієнті зашумованості σ = 10. У випадку, коли відстань Хаусдорфа між еталонними описами, що мають однакову назву, є найменшою у рядку, це означає, що еталонний опис було віднесено до правильного класу. На рисунку 3.7 можна побачити, що при запропонованому у якості прикладу рівні зашумованості, метрика Хаусдорфа впоралась з класифікацією на 100% правильно.

```
{
    'A - A': 88.0, 'A - B': 98.0, 'A - C': 96.0, 'A - D': 98.0, 'A - E': 95.0,
    'B - A': 97.0, 'B - B': 90.0, 'B - C': 99.0, 'B - D': 99.0, 'B - E': 99.0,
    'C - A': 97.0, 'C - B': 98.0, 'C - C': 89.0, 'C - D': 97.0, 'C - E': 97.0,
    'D - A': 97.0, 'D - B': 98.0, 'D - C': 97.0, 'D - D': 86.0, 'D - E': 98.0,
    'E - A': 98.0, 'E - B': 103.0, 'E - C': 96.0, 'E - D': 99.0, 'E - E': 90.0
}

# Method works correctly; 5 / 5 = 1.0
```

Рисунок 3.8 – Загальний результат порівняння еталонних описів з дескрипторами у бітовому виді при коефіцієнті зашумованості $\sigma = 10$

Тепер пропонується протестувати відстань Хаусдорфа з поданням у просторі ФУ. Для цього так само підійде скрипт, який було наведено у лістингу 3.8, але цього разу, будуть порівнюватись дескриптори, що представлені у цілочисельному вигляді, за допомогою Манхетенської метрики. Далі наведена програмна реалізація даної метрики.

Лістинг 3.9 Реалізація Манхетенської метрики:

```
def manhattan_distance(a, b):
    arr1 = np.array(a)
    arr2 = np.array(b)
    return np.sum(np.abs(arr1 - arr2))
```

У ході проведення експериментів, було встановлено, що відстань Хаусдорфа з поданням у просторі ФУ знаходиться в більш широкому інтервалі цілих значень [1960,…, 2034]. Комп’ютерний час для виконання обчислення метрики склав 29 секунд.

Ключовим моментом досліджень є ідея знаходження відстані Хаусдорфа з поданням у просторі 16 відібраних ФУ. Проведемо експеримент із використанням даної відстані. Для порівняння трансформованих дескрипторів, так само як і попередньо, використовується Мангетенська метрика. Отже, у результаті проведення експерименту, було виявлено, що відстань Хаусдорфа з поданням у просторі 16 відібраних ФУ знаходиться в

інтервалі цілих значень [126,…, 146]. У свою чергу, комп'ютерний час, який знадобився для виконання обчислення метрики, склав лише 1,7 секунди.

Далі, на рисунку 3.9 наведено загальну матрицю порівнянь усіх існуючих еталонних описів, яка було отримана у ході знаходження відстані Хаусдорфа з поданням у просторі 16 відібраних ФУ. Коефіцієнт зашумованості σ = 10.

```
{
    'A - A': 108, 'A - B': 126, 'A - C': 128, 'A - D': 130, 'A - E': 124,
    'B - A': 124, 'B - B': 112, 'B - C': 124, 'B - D': 132, 'B - E': 124,
    'C - A': 122, 'C - B': 126, 'C - C': 132, 'C - D': 134, 'C - E': 126,
    'D - A': 124, 'D - B': 134, 'D - C': 132, 'D - D': 112, 'D - E': 136,
    'E - A': 122, 'E - B': 126, 'E - C': 138, 'E - D': 122, 'E - E': 132
}

#   accurance = 3 / 5 = 0.6
```

Рисунок 3.9 – Загальний результат порівняння еталонних описів 16 ФУ при σ = 10

Орієнтуючись на рисунок, що наведено вище, можна зробити висновок, що цього разу, за того самого рівня зашумованості еталонних описів, який використовувався при порівнянні еталонних описів з дескрипторами у бітовому виді, можна зробити висновок, що точність класифікації впала, але час, який був витрачений на виконання класифікації, скоротився, приблизно, у 16 разів.

У ході експериментального вивчення завадостійкості розроблених підходів з використанням відстані Хаусдорфа при дії адитивного шуму було з'ясовано, що показник точності pr, який обчислюється відношенням числа правильно класифікованих об'єктів r_p до загального їх числа r, для традиційного методу і для методу з повним набором 256-ти ФУ дорівнює 1 за

умови, що $\mu < 8$, а при $\mu = 6$ – знижується до 0,8. У свою чергу, для методу із стисненим поданням 16 ФУ показник точності знижується до 0,9 при $\mu = 25$.

Далі наведено реалізацію методу, який відповідає за встановлення точності класифікації при роботі з метрикою Хаусдорфа.

Лістинг 3.10 Метод для знаходження точності класифікації у ході використання відстані Хаусдорфа:

```
def get_hausdorf_accurance(etalons, noisy_etalons, names, method, dict):
    sum = 0
    for i in range(len(etalons)):
        index, dict =
get_index_of_closest_etalon_for_etalon_by_hausdorf(etalons[i], noisy_etalons,
names[i], names, method, dict)
        print(f'Index of closest: {index}')
        if index == i:
            sum = sum + 1
    return sum/len(etalons), dict
```

Використовуючи метрику Хаусдорфа, можна побачити, що відстані, зазвичай, знаходяться у вузькому інтервалі, а найменша відстань між еталонними описами лише трохи відрізняється від загальної маси. Через це, показники точності класифікації, при збільшенні рівня зашумованості вхідних зображень, зменшуються. Саме тому, ідея використання відстані Хаусдорфа для порівняння еталонних описів виглядає ненадійною.

3.4 Аналіз впровадження відстані Танімото

Тепер черга застосувати метрику Танімото для множин у класифікаторі для трансформованого простору даних з використанням порогу (2.11) $\rho_{\lim}$ як 25% від максимального значення метрики для дескрипторів на усій множині еталонних даних. Важливим моментом є той факт, що, при цьому, максимальне значення метрики та поріг $\rho_{\lim}$ залежать від впровадженого простору даних.

Важливо відмітити, що відстань Танімото між звичайним еталоном і зашумованим поступово зростає у залежності від збільшення рівня шуму (зменшенням μ), що здійснює вплив на точність класифікації.

Далі наведено програмну реалізацію відстані Танімото.

Лістинг 3.11 Реалізація відстані Танімото:

```
def is_many_contains_same_descriptor(array, descriptor):
    counter = 0
    for i in range(len(array)):
        if is_descriptors_equal(descriptor, array[i]):
            counter += 1
    return counter > 0

def get_amount_uniq_descriptors_from_many(array_A, array_B):
    counter = 0
    for i in range(len(array_A)):
        if not is_many_contains_same_descriptor(array_B, array_A[i]):
            counter += 1
    return counter

def get_amount_no_uniq_descriptors_from_many(array_A, array_B):
    counter = 0
    for i in range(len(array_A)):
        if is_many_contains_same_descriptor(array_B, array_A[i]):
            counter += 1
    return counter

def get_tonimoto_distance(etalon, etalon_name, etalon_to_compare,
etlaon_to_compare_name):
    distance_a_b = get_amount_uniq_descriptors_from_many(etalon, etalon_to_compare)
    distance_b_a = get_amount_uniq_descriptors_from_many(etalon_to_compare, etalon)
    intersection = get_amount_no_uniq_descriptors_from_many(etalon, etalon_to_compare)
    result = (distance_a_b + distance_b_a) / \
             ((len(etalon) + len(etalon_to_compare)) - intersection)
    return result
```

Далі, у якості прикладів (рис. 3.10 – 3.12), можна побачити декілька результатів порівнянь звичайних еталонних описів та зашумованих описів з коефіцієнтом зашумованості, що дорівнює 10 за допомогою відстані Танімото. Порівняння проводились за допомогою методу з повним набором 256-ти ФУ, методу із стисненим поданням 16 ФУ та традиційного методу без застосування апарату ФУ. Якщо відстань між еталонними описами, що мають однакову назву, є найменшою, тоді це означає, що класифікація елементу відбулась правильно.

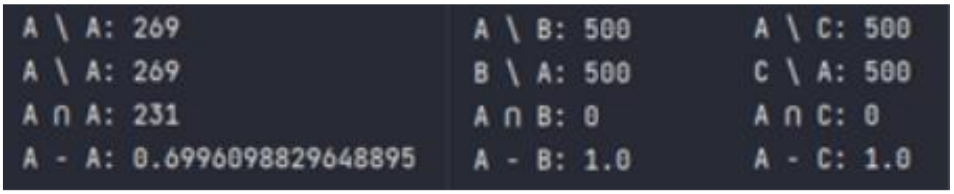


Рисунок 3.10 – Значення метрики Танімото для 256 ФУ при σ = 10

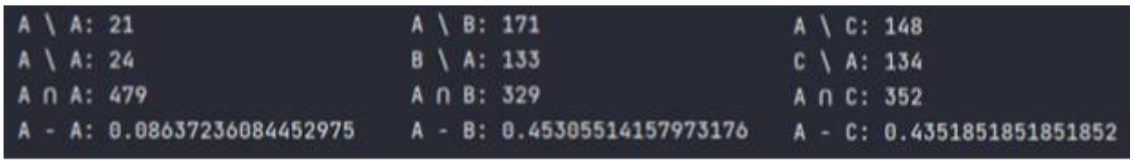


Рисунок 3.11 – Значення метрики Танімото для 16 ФУ при σ = 10

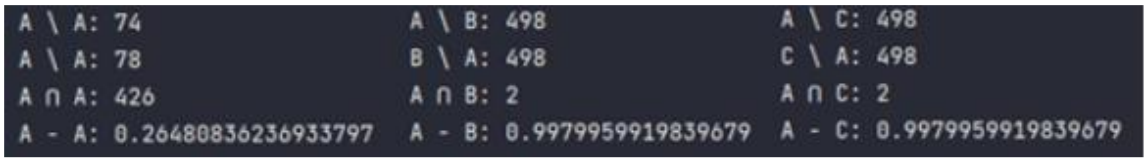


Рисунок 3.12 – Значення метрики Танімото без ФУ при σ = 10

У результаті проведення експериментального вивчення завадостійкості розроблених модифікацій класифікатора з використанням метрики Танімото і вибраного порогу ρ_{lim} під впливом адитивного шуму було встановлено, що показник (2.12) $pr(\mu)$ для методу з повним набором 256-ти ФУ, а також для методу із стисненим поданням 16 ФУ дорівнює 1 (класифікація без помилок) при $\mu \geq 1,2$ і знижується до 0,9 при $\mu = 1$ (рис. 3.13).

Лістинг 3.12 Реалізація методу встановлення точності класифікації у ході використання відстані Танімото:

```
def get_tonimoto_accurence(etalons, noisy_etalons, names, dict):
    sum = 0
    for i in range(len(etalons)):
            # index, dict = get_tonimoto_distance(etalons[i], names[i],
noisy_etalons[j], names[j], dict)
        index, dict = get_index_of_closest_etalon_for_etalon_by_tonimoto(etalons[i],
noisy_etalons, names[i], names,dict)
        print(f'Index of closest: {index}')
        if index == i:
            sum = sum + 1
        print(f'{sum} / {len(etalons)}\n')

    if sum == len(etalons):
```

```
            print(f'Method works correctly; {sum} / {len(etalons)} = {sum/len(etalons)}')
        else:
            print(f'{sum} / {len(etalons)} = {sum/len(etalons)}')

        return sum/len(etalons), dict
```

```
{
 'A - A': 0.086, 'A - B': 0.45, 'A - C': 0.43, 'A - D': 0.42, 'A - E': 0.38,
 'B - A': 0.47, 'B - B': 0.06, 'B - C': 0.42, 'B - D': 0.45, 'B - E': 0.459,
 'C - A': 0.41, 'C - B': 0.38, 'C - C': 0.07,'C - D': 0.4, 'C - E': 0.42,
 'D - A': 0.42,'D - B': 0.40, 'D - C': 0.42, 'D - D': 0.06, 'D - E': 0.42,
 'E - A': 0.38, 'E - B': 0.44, 'E - C': 0.43, 'E - D': 0.44, 'E - E': 0.05
}

# Method works correctly; 5 / 5 = 1.0
```

Рисунок 3.13 – Загальний результат порівняння еталонних описів 16 ФУ при σ = 10

Як можна побачити на рисунку 3.13, на відміну від відстані Хаусдорфа, метрика Танімото без проблем впоралась із задачею класифікації, не зважаючи на достатньо велику зашумованість вхідних зображень – σ = 10.

У свою чергу, використовуючи традиційний метод, у випадку без застосування апарату ФУ показник завадостійкості $pr(\mu)$ знаходиться практично в тих же межах. Для методу із стисненим поданням у 8 ФУ, з номерами 2,…,9 із таблиці 3.1 значення $pr(\mu)$ може знизитись до 0,9 при $\mu = 1,2$ (табл. 3.2). Варто відмітити, що дані показники завадостійкості є досить високими для прикладних систем, оскільки метод працює безпомилково навіть при відношенні сигнал-шум близьким до 1.

При цьому, відбувається пропорційне скорочення часу обчислення метрики (час класифікації) у модифікованих просторах даних до кількості використаних ФУ. У ході проведення експерименту було продемонстровано зниження часових витрат у порівнянні з повним набором 256-ти ФУ для 16-ти ФУ у 17 разів, а для 8-ми ФУ – у 29 разів (табл. 3.3). При цьому, час виконання класифікації для традиційного методу шляхом обчислення

метрики Танімото для описів без трансформації отримано у 11 разів більший, ніж для модифікації із 16 ФУ.

Таблиця 3.2 – Експериментальна оцінка точності класифікації у залежності від рівня шуму

Метод	СКВ						
	5	10	20	30	50	75	100
ФУ 256	1	1	1	1	1	1	1
ФУ 16	1	1	1	1	1	1	1
ФУ 8	1	1	1	1	1	0,98	0,91
Традиц.	1	1	1	1	1	1	1

Таблиця 3.3 – Експериментальна оцінка часу класифікації, с

Метод	Час
256 ФУ	222,98 с
16 ФУ	12,6 с
8 ФУ	7,834 с
Традиційний	138,92 с

Таким чином, використовуючи саме відстань Танімото, вдалось скоротити час виконання класифікації зображень у декілька разів. При цьому зберегти показники точності.

ВИСНОВКИ

У рамках кваліфікаційної роботи було розроблено і реалізовано удосконалення результативності структурних методів класифікації зображень шляхом впровадження апарату розкладання компонентів опису за системою ортогональних функцій Уолша.

Використовуючи систему ортогональних функцій для подання опису зображень як множини дескрипторів КТ вдалось отримати суттєвий виграш у швидкодії оброблення. При цьому, високі показники точності та завадостійкості класифікації збереглися.

Вибір порогу для встановлення еквівалентності компонентів та метрики для зіставлення трансформованих описів у новоствореному просторі даних – це ключовий момент ефективності такого подання. Виявилось, що ефективнішим є використання метрики Танімото за умови використання порогу як 25% від максимального значення вибраної метрики для еталонного набору даних.

Визначення та застосування складу скороченого кортежу ФУ для подання даних впливає на точність класифікації.

Дослідження показало, що впровадження апарату функцій Уолша дає можливість не лише скоротити обчислювальні витрати у десятки разів, але також дає змогу забезпечити високі показники результативності класифікації.

У свою чергу, практична значущість роботи полягає у тому, що були побудові моделі класифікації у трансформованому просторі даних. Була підтверджена працездатність та завадостійкість запропонованих модифікацій на прикладах зображень. Створено програмні застосунки з метою впровадження розроблених класифікаторів у системах комп’ютерного зору.

Таким чином, перспективи дослідження можуть бути пов’язані із опрацюванням та аналізом ефективності для різноманіття моделей формування стиснених описів.

Результати дослідження апробовано у вигляді статті, опублікованої у науково-технічному журналі «СУЧАСНІ ІНФОРМАЦІЙНІ СИСТЕМИ» [43].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Ahmed N., and Rao K.R. (1975), Orthogonal Transforms for Digital Signal Processing, Springer Ferlag Berlin, 248 p.

2. Scherer, R., Computer Vision Methods for Fast Image Classification and Retrieval, Studies in Computation Intelligence 821, Springer Nature Switzerland AG 2020. https://pdfcoffee.com/computer-vision-methods-for-fast-image-classification-and-retrieval-2020-pdf-free.html.

3. Pratt W. K. (2001), Digital Image Processing, New York: John Wiley and Sons Inc., 723 p.

4. Vlasenko, N. V., & Sytnik, O. (2013). Classification of video-objects in attribute space of the Walsh functions. *Telecommunications and Radio Engineering*, 79(10), pp. 1777–1785.

5. Flach, P. (2012). *Machine learning: the art and science of algorithms that make sense of data*. Cambridge university press: New York, NY, USA, 409 p.

6. Alessio, S. M. (2015). Digital signal processing and spectral analysis for scientists: concepts and applications, 200 p.

7. Gorokhovatskiy, V. A. (2011). Compression of descriptions in the structural image recognition. *Telecommunications and Radio Engineering*, 70(15), pp. 1363–1371.

8. 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, pp. 58–68.

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

10. Gorokhovatskyi, V., Gadetska, S., Ponomarenko, R. (2020). Recognition of Visual Objects Based on Statistical Distributions for Blocks of

Structural Description of Image. Lecture Notes in Computational Intelligence and Decision Making. Proceedings of the XV International Scientific Conference "*Intellectual Systems of Decision Making and Problems of Computational Intelligence*" (ISDMCI'2019), Ukraine, May 21–25, 2019, pp. 501–512.

11. Shapiro, L., (2001). Computer vision. Prentice Hall, 625 p.

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

13. Гороховатський, В.О., Власенко Н.В. (2021). Редукція опису зображення у складі множини дескрипторів на основі метричного критерію інформативності. Advanced Information Systems, 5(4), pp. 10–16.

14. Гороховатский В.А. (2003) Распознавание изображений в условиях неполной информации. Харків: ХНУРЭ, 112 с.

15. Гороховатський В.О., Творошенко І.С. (2022) Аналіз багатовимірних даних за описом у формі множини компонент: монографія. Харків: ХНУРЕ, 124 с.

16. Колмогоров А.Н., Фомин С.В. (2022). Элементы теории функций и функционального анализа.

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

18. Tvoroshenko I., and Gorokhovatskyi V. (2022) The Application of Hybrid Intelligence Systems for Dynamic Data Analysis, International Journal of Engineering and Information Systems, 6(2), pp. 40 –48.

19. Gadetska S., Gorokhovatskyi V., Stiahlyk N., Vlasenko N. (2022) Aggregate Parametric Representation of Image Structural Description in Statistical Classification Methods. In CEUR Workshop Proceedings: Computer Modeling and Intelligent Systems (CMIS-2022), 3137, pp. 68–77.

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

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

22. Гороховатський, В.О., Гадецька, С.В. (2020). Статистичне оброблення та аналіз даних у структурних методах класифікації зображень (монографія), Харків, ФОП Панов А.Н., 128 с.

23. Загоруйко, Н. Г. (1999). Прикладные методы анализа данных и знаний.

24. Залманзон, Л. А. (1989). Преобразования Фурье, Уолша, Хаара и их применение в управлении, связи и других областях. Наука, 496 с.

25. Ересько, Ю. Н. (2002). Локализация изображений в автоматических визирах. М.: Компания Спутник, 357 с.

26. Gorokhovatskyi, V., Rusakova, N., Tvoroshenko, I. (2020) The application of image analysis methods and predicate logic in applied problems of magnetic monitoring. Telecommunications and Radio Engineering, 79 (20), pp. 1801–1811.

27. Gorokhovatskiy, V. A., & Putyatin, Y. P. (2009). Image Likelihood Measures on the Basis of the Set of Conformities. Telecommunications and Radio Engineering, 68(9).

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

29. Чмутов Ю.В. (2021) Методи швидкої класифікації зображень на підставі кластерезації: матеріали конференції «Сучасні методи обробки зображень». (Харків, 20-22 квітня 2021 р.). Харків, С 12–13.

30. Kohonen, T. (2001). Self-Organizing Maps. Springer-Verlag, Berlin, Heidelberg doi:book/10.5555/558021.

31. Gorokhovatsky V.A., Putyatin Ye. P. (2009). Image Likelihood Measures of the Basis of the Set of Conformities. Telecommunications and Radio Engineering. 68 (9), pp. 763–778.

32. Kinoshenko, D., Mashtalir, V., Yegorova, E., & Vinarsky, V. (2005, July). Hierarchical partitions for content image retrieval from large-scale database. In International Workshop on Machine Learning and Data Mining in Pattern Recognition. Springer, Berlin, Heidelberg, pp. 445–455.

33. M. A. Ahmad, V. Gorokhovatskyi, I. Tvoroshenko, N. Vlasenko, S. K. Mustafa (2021) The Research of Image Classification Methods Based on the Introducing Cluster Representation Parameters for the Structural Description, International Journal of Engineering Trends and Technology, 69(10), pp. 186–192.

34. Walsh, J. L. (1923). A closed set of normal orthogonal functions. American Journal of Mathematics, 45(1), 5–24.

35. Lackey, R. B., & Meltzer, D. (1971). A simplified definition of Walsh functions. IEEE Transactions on Computers, 100(2), 211–213.

36. Paley, R. E. A. C. (1931). A remarkable series of orthogonal functions. Proc. London Math. Soc, 34(1), 241–279.

37. Yuen, C. (1971). Walsh functions and Gray code. IEEE Transactions on Electromagnetic Compatibility, (3), 68–73.

38. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., and Al-Dhaifallah M. (2022). Classification of Images Based on a System of Hierarchical Features, Computers, Materials & Continua, 72(1), pp. 1785–1797, DOI: 10.32604/cmc.2022.025499.

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

40. Гороховатський В.О., Пупченко Д.В., Солодченко К.Г. (2018) Аналіз властивостей, характеристик та результатів застосування новітніх детекторів для визначення особливих точок зображення. Системи управління, навігації та зв’язку. С. 93–98.

41. Гороховатский В.А., Передрий Е.О. (2009) Корреляционные методы распознавания изображений путем голосования систем фрагментов. Радіоелектроніка, інформатика, управління, №1 (20), с.74–81.

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

43. Гороховатський, В.О., Творошенко, І.С., Чмутов, Ю.В. (2022). Застосування систем ортогональних функцій для формування простору ознак у методах класифікації зображень. Сучасні інформаційні системи, т. 6, №3, с. 5–12.

44. Гороховатський В., Творошенко І., Сидоренко Д. (2021). Класифікація зображень із використанням кластерного подання, Міжнародний науковий симпозіум «Інтелектуальні рішення-С». Обчислювальний інтелект (результати, проблеми, перспективи). Теорія прийняття рішень: праці міжн. наук. симпозіуму (Вересень 29, 2021). Київ – Ужгород, С. 44–45.

45. Smeulders, A. W., Worring, M., Santini, S., Gupta, A., & Jain, R. (2000). Content-based image retrieval at the end of the early years. IEEE Transactions on pattern analysis and machine intelligence, 22(12), 1349–1380.

46. Berman, A. P., & Shapiro, L. G. (1999). A flexible image database system for content-based retrieval. Computer Vision and Image Understanding, 75(1-2), 175–195.