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)

ДОСЛІДЖЕННЯ МЕТОДІВ РОЗПІЗНАННЯ ТА ВИЗНАЧЕННЯ СТУПЕНЮ СХОЖОСТІ ОБЛИЧ
(тема)

Виконав:
студент 2 курсу, групи ІНФм-22-2
Тарасов Д.К.
(прізвище, ініціали)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Творошенко І.С.
(посада, прізвище, ініціали)

Допускається до захисту

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

2024 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)
«____»_____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Тарасову Данилу Кириловичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження методів розпізнавання та визначення ступеню схожості облич.

затверджена наказом по університету від 3 листопада 2023 року № 1280Ст

2. Термін подання студентом роботи до екзаменаційної комісії 24 грудня 2023 р.

3. Вихідні дані до роботи математичні моделі визначення ступеню схожості облич, теоретичні відомості про методи вилучення ознак облич на текстурних зображеннях.

4. Перелік питань, що потрібно опрацювати в роботі

1. Огляд методів розпізнання та визначення ступеню схожості облич.
2. Опис нейронних мереж та детекторів облич, яка використана в роботі.
3. Програмна реалізація застосунку для розпізнання та визначення ступеня схожості облич.
4. Аналіз кінцевих результатів та визначення перспектив подальшої роботи.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність проблеми визначення ступеню схожості облич, постановка задачі дослідження, аналіз предметної області, вихідні дані до дослідження, етапи реалізації задачі, аналіз результатів.

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

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	03.11.2023	
2	Аналіз завдання, підбір літератури	03.11.23-04.11.23	
3	Аналіз літератури з досліджуваної проблеми	04.11.23-07.11.23	
4	Аналіз методів розпізнавання	08.11.23-12.11.23	
5	Аналіз детекторів облич	13.12.23-14.12.23	
6	Програмна реалізація	15.12.23-21.12.23	
7	Оформлення пояснювальної записки	21.12.23-24.12.23	
8	Перевірка на плагіат	30.11.2023	
9	Рецензування	10.12.2023	
10	Підготовка презентації та доповіді	16.12.2023	
11	Занесення роботи в електронний архів	23.12.2023	
12	Попередній захист кваліфікаційної роботи	02.01.2024	

Дата видачі завдання 3 листопада 2023 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Творошенко І.С.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 88 с., 21 табл., 36 рис., 1 дод., 45 джерела.

ТЕХНОЛОГІЯ РОЗПІЗНАВАННЯ ОБЛИЧ, АНАЛІЗ ГОЛОВНИХ КОМПОНЕНТІВ, ГЛИБОКІ ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ, ЛОКАЛЬНІ БІНАРНІ ШАБЛОНИ, ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ, ВІЗУАЛЬНА ГЕОМЕТРИЧНА ГРУПА.

Об’єктом дослідження є розпізнавання та визначення ступеню схожості облич.

Метою дослідження є вивчення та порівняння методів для визначення ступеню схожості облич.

Використано методи розпізнавання облич та визначення ступеню схожості облич. Проведено дослідження методів порівняння зображень та визначення ступеню їх схожості, детектування та аналіз облич на основі сучасних нейронних мереж DeepFace, Facenet, Openface та VGGNet. Досліджено методи детектування схожості облич та розроблено алгоритм визначення ступеню схожості облич.

У результаті дослідження здійснена програмна реалізація системи для порівняння облич та визначення їх ступеню схожості.

FACE RECOGNITION TECHNOLOGY, PRINCIPAL COMPONENT ANALYSIS, DEEP CONVOLUTIONAL NEURAL NETWORK, LOCAL BINARY PATTERN, CONVOLUTIONAL NEURAL NETWORK, VISUAL GEOMETRY GROUP.

The object of the research is the recognition and determination of the degree of facial similarity.

The aim of the research is to investigate and compare methods for determining the degree of similarity of faces.

Methods used face recognition and determination of the degree of similarity of faces. Research methods of comparing images and determining the degree of their similarity, detection and analysis of faces based on modern neural networks DeepFace, Facenet, Openface and VGGNet. The method of detecting the similarity of faces was studied and an algorithm for determining the degree of similarity of faces was developed.

As a result of implemented software implementation of the system to for comparing faces and determining their degree of similarity.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

DARPA – Defense Advanced Research Projects Agency (агентство передових оборонних дослідницьких проєктів США)

США – Сполучені Штати Америки

FERET – Face Recognition Technology (технологія розпізнавання облич)

PCA – Principal Component Analysis (аналіз головних компонентів)

CNN – Convolutional Neural Network (згорткові нейронні мережі)

VGG – Visual Geometry Group (візуальна геометрична група)

BESNet – Boundary Extraction Constrained Siamese Network (сіамська мережа з обмеженим вилученням)

MSBE – MultiScale Boundary Extraction (багатомасштабне виділення меж)

BEC – Boundary Extraction Constrained (обмежене виділення межі)

CD – Change Detection (виявлення змін)

LFW – Labeled Faces in the Wild (база даних маркованих облич в дикій природі)

HFR – Human Face Recognition (система розпізнавання обличчя людини)

DCNN – Deep Convolutional Neural Network (глибокі згорткові нейронні мережі)

LDA – Linear Discriminant Analysis (лінійний дискримінантний аналіз)

LPP – Local Preserving Projections (проєкція зі збереженням лінійності)

LBP – Local Binary Pattern (локальні бінарні шаблони)

SSD – Single Shot-Multibox Detector (детектор осередків одного знімка)

ВСТУП

Сучасний розвиток технологій та інформаційних систем допомагає життю людей та суспільства в цілому, надаючи можливості для автоматизації та полегшення рутинних завдань. Однією з найбільш популярних галузей в цьому контексті є обробка зображень та комп'ютерне бачення. Серед цікавих завдань у цій області є розпізнавання та визначення ступеня схожості досліджених об'єктів, наприклад людських облич.

Обличчя людини – це важлива складова її ідентичності, вона відіграє вирішальну роль в багатьох аспектах життя: від біометричної ідентифікації та безпеки до соціальної взаємодії та реклами. Слід зазначити, що розпізнавання та визначення схожості облич має великий потенціал і застосовується у різних сферах, таких як відеоспостереження, медицина, емоційний аналіз, віртуальна реальність та багато інших.

Для вирішення проблем розпізнавання досліджених об'єктів на сьогодні, наприклад, нейронні мережі здатні реалізувати велику частину різних завдань, таких як правильний аналіз світла навколо обличчя людини, коректне розпізнавання об'єктів на обличчі та розпізнавання положення обличчя.

Для оцінки ефективності відомих методів розпізнавання осіб, агентство DARPA та дослідницька лабораторія армії США розробили програму FERET (Face Recognition Technology) [1]. У масштабних тестах програми FERET брали участь підходи, що ґрунтуються на гнучкому порівнянні на графах та всілякі модифікації методу головних компонентів (Principal Component Analysis).

Актуальність дослідження полягає в порівнянні методів розпізнавання, визначення ступеню схожості облич та виявлення оптимального методу.

Результати дослідження порівняння методів розпізнавання облич та визначення ступеню їх схожості мають велике значення у різноманітних галузях як безпека та ідентифікація осіб, робототехніка та штучний інтелект.

1 АНАЛІЗ ІСНУЮЧИХ ЗАСТОСУНКІВ ДЛЯ РОЗПІЗНАВАННЯ ТА ВИЗНАЧЕННЯ СТУПЕНЮ СХОЖОСТІ ОБЛИЧ

1.1 Аналіз сучасних застосунків для розпізнавання та визначення ступеню схожості облич

Із швидким розвитком глибокого навчання та обробки зображень, розпізнавання та визначення схожості облич стали актуальними завданнями, що знаходять застосування у різних галузях. Ця область пройшла значний розвиток завдяки інноваціям у глибокому навчанні, та сьогодні нараховує велику кількість сучасних застосунків, які використовуються для розпізнавання облич та визначення ступеню їх схожості.

Визначення та розпізнавання облич стали не лише ключовими елементами численних технологій, але й відіграють важливу роль у суспільстві та бізнесі. Від використання розпізнавання облич для забезпечення безпеки та доступу до систем до застосування в рекламі та медицині ці технології знаходять широке застосування та активно розвиваються.

Під час моніторингу, пошуку та збору відповідної інформації про цільову особу, традиційні ручні методи піддаються впливу різних факторів, таких як людська увага, спостереження, наявність знань та особиста оцінка. Це може спричиняти недоліки та помилки через людський фактор. Тут на допомогу приходить технологія розпізнавання обличчя, яка об’єднує комп’ютерну обробку зображень та методи біостатистики.

Технологія розпізнавання обличчя використовує комп’ютерну обробку зображень для виділення основних рис обличчя з відеоматеріалів, а потім застосовує принципи біостатистики для створення математичної моделі [2]. У порівнянні з іншими біометричними технологіями, технологія розпізнавання обличчя вирізняється своєю надійністю, точністю та іншими перевагами.

Основою технології розпізнавання обличчя є великий обсяг відеоданих та оптимізований метод глибокого навчання. Цей спосіб здатний автоматично

збирати, порівнювати та шукати обличчя в інтелектуальний спосіб. Таким чином, технологія розпізнавання обличчя здатна інтелектуально, ефективно та швидко збирати, фіксувати та обробляти інформацію про обличчя, що дозволяє досягти мети ідентифікації в реальному часі та ефективно боротися з підозрюваними [3].

Сучасні нейронні мережі для розпізнавання та визначення ступеня схожості облич надзвичайно потужні та датуються останніми досягненнями в глибокому навчанні та обробці зображень. Вони використовуються в різних застосунках для вирішення задач ідентифікації та верифікації облич.

FaceNet – це глибока нейронна мережа для розпізнавання облич, розроблена дослідниками компанії Google у 2015 році. FaceNet є однією з перших нейронних мереж, яка досягла вражаючих результатів у завданні розпізнавання облич і визначення ступеня їх схожості. Основна ідея застосування FaceNet полягає в створенні векторних представлень облич, які можуть бути використані для порівняння і ідентифікації осіб [4].

Основні особливості FaceNet:

– Triplet Loss (потрійне навчання): FaceNet використовує спеціальну функцію втрат, відому як Triplet Loss, для навчання мережі. Ця функція втрат бере трійки зображень: як мінімум два з них належать одній і тій же особі (anchor та positive), і третє зображення належить різній особі (negative). У результаті, мережа навчається так, щоб векторні подання облич певної особи були ближче один до одного, ніж до представлень інших осіб;

– векторні подання облич: FaceNet створює векторні подання (або ембедінги) для кожного обличчя на зображенні. Ці вектори мають фіксований розмір та властивість, що відображає семантичну схожість між обличчями. Ближчі вектори відповідають більш схожим обличчям, а далекі вектори – менш схожим;

– застосування згорткових нейронних мереж (Convolutional Neural Network, CNN): FaceNet використовує згорткові нейронні мережі для

вилучення важливих ознак з облич на зображеннях. Ці мережі спеціально адаптовані для розпізнавання облич;

– використання в сфері розпізнавання облич: FaceNet застосовується для ідентифікації осіб на фотографіях, відеозаписах і навіть в реальному часі в системах безпеки, банківських застосунках та соціальних мережах. FaceNet був одним із перших кроків у використанні глибокого навчання для розпізнавання облич. З того часу були розроблені інші моделі, але FaceNet продовжує бути важливим внеском у цю область, завдяки своїм інноваційним методам навчання та вражаючій точності.

DeepFace – це інноваційна система розпізнавання обличчя, розроблена командою вчених у Facebook (тепер Meta Platforms, Inc.), яка була представлена у 2014 році. DeepFace відзначається високою точністю та ефективністю у завданні розпізнавання облич, і вона використовується в соціальних мережах та інших застосунках для ідентифікації осіб на фотографіях [5].

Основні характеристики DeepFace:

– згорткові нейронні мережі (CNN): DeepFace використовує глибоку згорткову нейронну мережу для аналізу зображень обличчя. Ця мережа автоматично виявляє та екстрагує важливі ознаки з облич, такі як очі, ніс, рот, які використовуються для подальшої ідентифікації;

– людино-подібна точність: DeepFace досягла вражаючої точності розпізнавання облич, що наближається до людського рівня. За допомогою великої кількості даних для навчання і технологій глибокого навчання, вона здатна надзвичайно точно розпізнавати та ідентифікувати обличчя на фотографіях;

– орієнтаційна нечутливість: DeepFace може розпізнавати обличчя, незалежно від їх орієнтації та положення на зображенні. Ця властивість робить її досить робочою і надійною для різних умов і фотографій;

– застосування в соціальних мережах: DeepFace впроваджено в соціальних мережах, таких як Facebook, для автоматичної ідентифікації

користувачів на фотографіях та підтримки автоматичного тегування осіб на зображеннях.

DeepFace була важливим кроком у напрямку розпізнавання облич та вдалого використання глибокого навчання для цієї задачі. Ця технологія продовжує розвиватися і знаходити застосування у багатьох областях, де потрібна точна ідентифікація облич.

ArcFace – це інноваційний метод для розпізнавання облич і визначення ступеня схожості між ними, оснований на глибокому навчанні. Цей підхід був представлений дослідниками в 2019 році та широко використовується в системах біометричної ідентифікації та верифікації осіб. ArcFace вирізняється своєю високою точністю та стійкістю до різних змін, таких як освітлення та поза об’єкта.

Основні характеристики ArcFace:

– ArcMargin Loss: основним інноваційним компонентом ArcFace є використання ArcMargin Loss (або Angular Margin Loss) для навчання моделі. Ця функція втрат враховує ангулярний відхил між векторами представлення облич та застосовує маржі між ними. Це допомагає збільшити відстань між векторами представлення облич, що належать різним особам, та зменшити відстань між векторами одного обличчя;

– векторні представлення облич: так само, як і в інших методах глибокого навчання для розпізнавання облич, ArcFace створює векторні представлення (або ембедінги) для кожного обличчя на зображенні. Ці вектори представлення мають фіксований розмір і є унікальними для кожного обличчя;

– семантична схожість: векторні подання облич, створені ArcFace, дуже чутливі до семантичної схожості між обличчями. Це означає, що вони добре відділяють одних осіб від інших навіть при наявності схожих фізичних ознак;

– застосування в біометриці: ArcFace використовується для ідентифікації та верифікації осіб в системах безпеки, банківських застосунках, відеоспостереженні, а також в інших застосунках, де важлива точність та надійність розпізнавання облич;

– стійкість до змін: ArcFace виявляє стійкість до різних умов зйомки, таких як освітлення, кути зйомки та інші фактори, що допомагає досягти високої точності в різних умовах.

ArcFace є одним із передових методів у сфері розпізнавання облич і використовується у великому ряді застосунків для біометричної ідентифікації та верифікації осіб [6].

OpenFace – це відкрите програмне забезпечення для розпізнавання облич та визначення схожості облич, розроблене дослідниками з компанії Carnegie Mellon University. Це програмне забезпечення використовує глибокі нейронні мережі для створення векторних представлень облич та подальшого порівняння їх для визначення ступеня схожості [7].

Основні особливості OpenFace:

– детекція та локалізація облич: OpenFace використовує згорткові нейронні мережі для виявлення та локалізації облич на вхідних зображеннях. Цей процес включає в себе визначення координат облич та орієнтації облич на зображенні;

– векторні представлення облич: після детекції облич OpenFace створює векторні представлення облич для кожного виявленого обличчя. Ці вектори представлення є унікальними для кожного обличчя і використовуються для подальшого порівняння та ідентифікації;

– легкість використання: OpenFace надає інтерфейс командного рядка та бібліотеки Python для використання. Це робить його доступним та легким у використанні для дослідників та розробників, які хочуть використовувати розпізнавання облич у своїх застосунках чи дослідженнях;

– відкритий код: OpenFace є відкритим програмним забезпеченням і доступний для безкоштовного використання та розповсюдження. Це робить його доступним для розробників і дослідників з усього світу для використання та покращення;

– застосування: OpenFace може бути використаний у багатьох сферах, включаючи системи безпеки, біометричну ідентифікацію, автоматизоване

тегування фотографій, аналіз емоцій та багато інших застосунків, де необхідно розпізнавання облич та визначення схожості.

VGGFace – це набір моделей глибокого навчання, розроблених на основі архітектури Visual Geometry Group (VGG) для розпізнавання облич та векторного представлення облич. Ці моделі були розроблені дослідниками університету Оксфорд та представлені для використання у завданнях розпізнавання та верифікації облич на зображеннях [8].

Основні особливості VGGFace:

– основано на VGGNet: Моделі VGGFace базуються на архітектурі VGGNet, яка вперше була розроблена для завдань класифікації зображень. VGGNet популярна своєю глибокою архітектурою з багатьма шарами згортки, що допомагає здійснювати відмінну вилучення ознак інформації з зображень;

– векторні представлення облич: VGGFace генерує векторні представлення (або ембедінги) для облич на зображеннях. Ці вектори представлення мають фіксований розмір і використовуються для порівняння та ідентифікації облич;

– ідентифікація та верифікація: VGGFace може використовуватися як для завдань ідентифікації, де потрібно визначити, які особи зображені на фотографії, так і для завдань верифікації, де визначається, чи належить обличчя на фотографії певній особі;

– застосування: VGGFace знаходить застосування в різних сферах, включаючи біометричну ідентифікацію, системи безпеки, автоматизоване тегування фотографій, аналіз емоцій, системи контролю доступу і багато інших областей, де необхідно розпізнавання облич. Моделі VGGFace демонструють високу точність в розпізнаванні облич та ідентифікації осіб;

– доступність: Моделі VGGFace доступні для безкоштовного використання та досліджень, що робить їх популярними в галузі глибокого навчання та розпізнавання облич.

Таким чином, кожна з вище перерахованих нейронних мереж має свої особливості та переваги, які використовуються у різних областях, наприклад, таких, як у безпеці, біоінженерії та соціальних мережах.

1.2 Класифікація та аналіз існуючих методів для розпізнавання та визначення ступеню схожості облич

У сучасному світі, де технології знаходяться на піку розвитку, розпізнавання та визначення ступеню схожості облич стали актуальною темою, яка знаходить своє застосування в різних сферах нашого життя. Від безпеки та біометричних ідентифікаційних систем до розваг та досліджень в галузі штучного інтелекту, технології розпізнавання облич стали невід'ємною частиною нашої сучасної реальності.

Застосунки для розпізнавання облич здатні визначати та ідентифікувати осіб, враховуючи унікальні особливості. Ця технологія включає в себе використання різних методів, від класичних до сучасних, що базуються на глибокому навчанні та штучних нейронних мережах.

Серед найпопулярніших та найефективніших можна виділити: SNN – Siamese Neural Networks та Eigenfaces.

1.2.1 SNN

Наразі існує декілька моделей глибокого навчання, таких як BERT, GAN і U-Nets, які забезпечують найсучасніше виконання таких завдань, як розпізнавання зображень, сегментація зображень і моделювання мови. Навряд чи проходить день без нових інновацій у машинному навчанні [9].

Такі технічні гіганти, як Google, Microsoft і Amazon, розробляють складні архітектури глибокого навчання, які забезпечують продуктивність, схожу на людську. Але одна проблема з цими моделями полягає в тому, що

вони вимагають великих даних з мітками. Іноді багато даних недоступні для конкретного завдання. Менше даних означає, що модель глибокого навчання не зможе належним чином моделювати різні класи та працюватиме погано. На допомогу приходять сіамські мережі, які допомагають створювати моделі з високою точністю навіть на невеликій кількості вибірок на клас і незбалансованим розподілом класів.

Сіамська мережа – це клас нейронних мереж, який містить одну або кілька ідентичних мереж. Подаємо пару входів до цих мереж, зображених на рисунку 1.1.

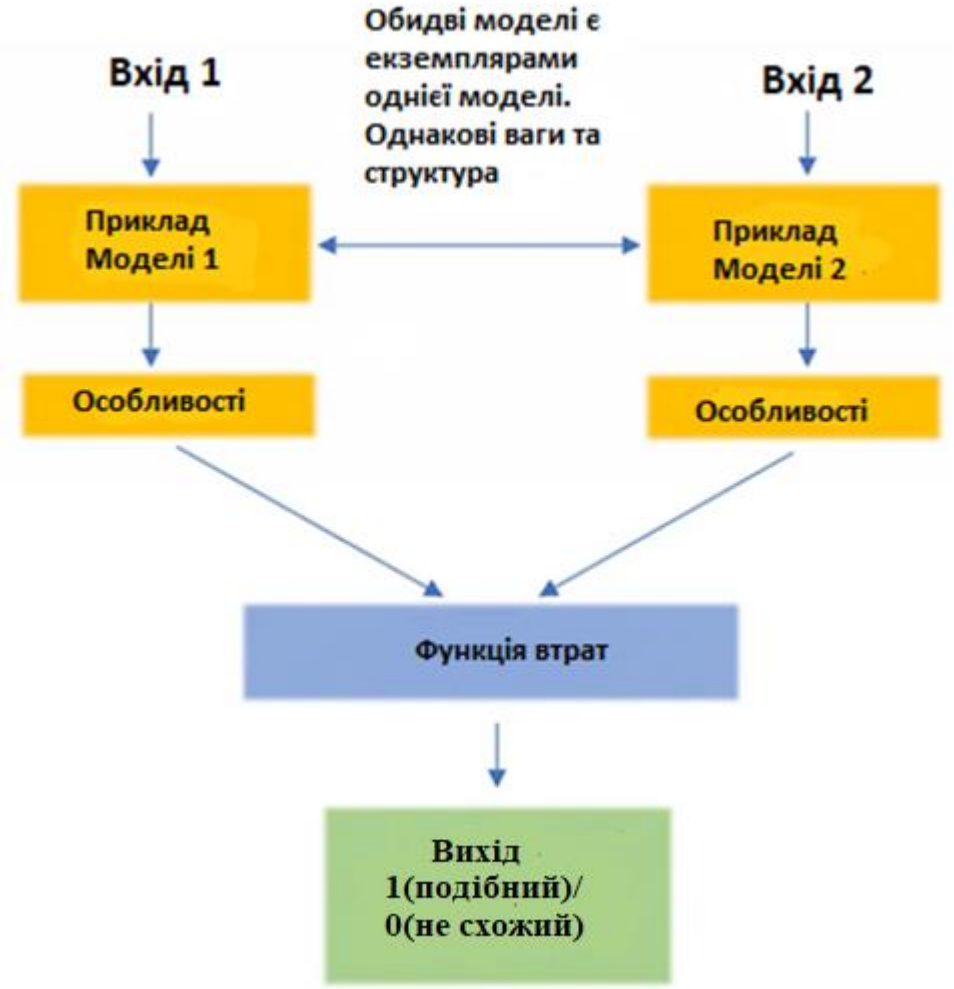


Рисунок 1.1 – Базова структура сіамської мережі

Кожна мережа обчислює характеристики одного входу. Потім подібність ознак обчислюється за допомогою їх різниці або скалярного добутку. Для однакових вхідних пар класів цільовий вихід дорівнює 1, а для різних вхідних пар класів вихід дорівнює 0. Обидві мережі мають однакові параметри та вагу. Якщо ні, то вони не сіамці [10].

Роблячи це, перетворюємо проблему класифікації на проблему подібності. Навчаємо мережу мінімізувати відстань між зразками одного класу та збільшити відстань між класами. Існує кілька видів функцій подібності, за допомогою яких можна навчити сіамську мережу, як втрата контрасту та втрата триплету.

Функція контрастних втрат. Сіамська мережа призначена не для виконання класифікації вхідних зображень, а для розрізнення вхідних зображень. Таким чином, функції втрати класифікації, такі як втрата перехресної ентропії, не підійдуть найкраще. Натомість ця сіамська мережева архітектура краще підходить для використання контрастної функції [11]. Ця функція просто оцінює, наскільки добре сіамська мережа здатна розрізняти задані пари зображень, схема зображена на рисунку 1.2.

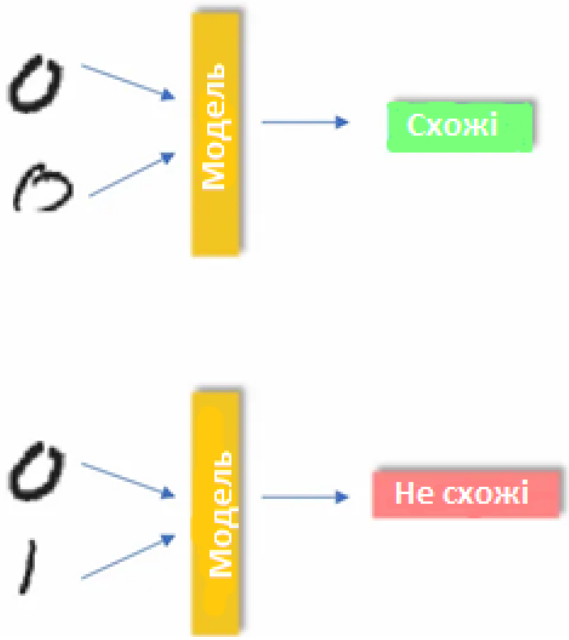


Рисунок 1.2 – Втрата контрасту: у верхній частині зображено однакові зображення класу; у нижній – різні зображення

Формула контрастної функції втрат має такий вигляд:

$$(1 - Y)\frac{1}{2}(D_w)^2 + (Y)\frac{1}{2}\{\max(0, m - D_w)\}^2, \quad (1.1)$$

де D_w – евклідова відстань між виходами сіамських мереж;

Y дорівнює 1 або 0. Якщо перше та друге зображення належать до одного класу, значення *Y* дорівнює 0, інакше *Y* дорівнює 1;

max() – це функція, яка позначає вище значення між 0 і m-D_w;

m – це значення маржі, яке перевищує 0. Наявність маржі означає, що різнорідні пари, які виходять за межі цієї маржі, не сприятимуть збитку.

Триплетні втрати дозволять нашій моделі відобразити дві подібні зображення, близькі та далекі від різних пар зразкових зображень [12]. Цей метод реалізовано за допомогою триплету, зображеного на рисунку 1.3, який містить:

– зображення прив’язки – це зразок зображення;

– позитивне зображення – це ще один варіант опорного зображення. Такий підхід допомагає моделі сіамської мережі дізнатися схожість між двома зображеннями;

– негативне зображення – це зображення, відмінне від наведених вище двох подібних пар зображень.

Описаний спосіб допомагає моделі вивчати відмінності за допомогою опорних зображень. Тепер, щоб збільшити відстань між подібними та несхожими парами вихідних векторів і відобразити подібні зображення близько одне до одного, існує термін, відомий як запас. Маржа збільшує розділення між нашим подібним і несхожим вектором, а також усуває результат будь-якого тривіального рішення.

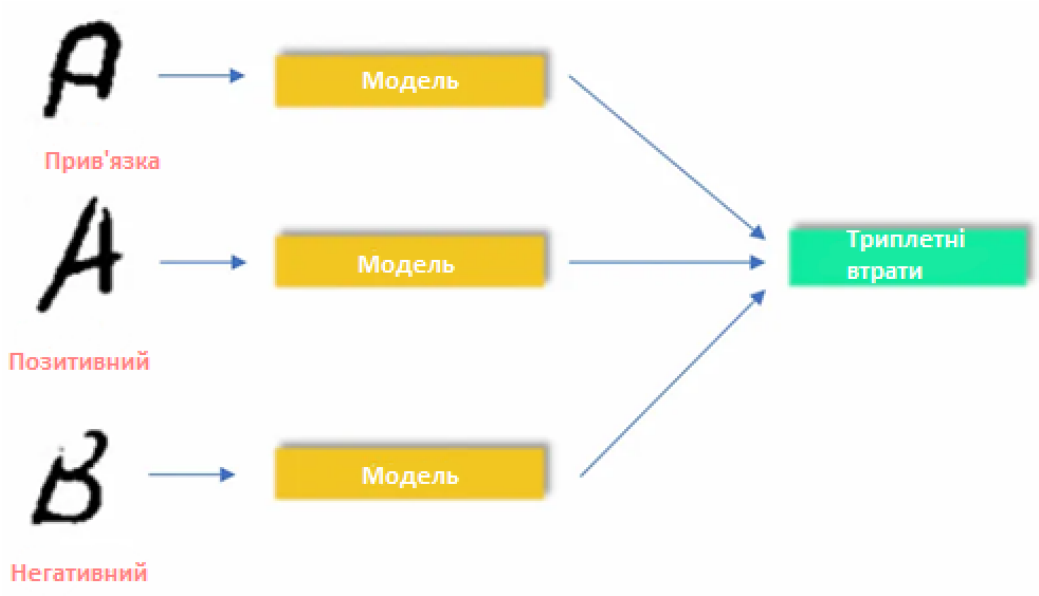


Рисунок 1.3 – Реалізація триплетних втрат

Функція втрат:

$$loss(a, p, n) = \max(0, d(a, p) - d(a, n) + margin), \quad (1.2)$$

де a – опорне зображення;

p – позитивне зображення;

n – негативне зображення.

Таким чином, сіамські мережі допомагають моделям вивчати різниці між зображеннями за допомогою опорних зображень та функцій триплетних втрат. Однак, даний вид мереж не використовується для глибокої класифікації зображень.

1.2.2 Eigenfaces

Основою методу власних граней є аналіз головних компонентів (Principal Component Analysis). Власні обличчя та PCA були використані Сировичем Лоуренсом та Кірбі Майклом для ефективного представлення зображень обличчя.

Вони почали з групи оригінальних зображень обличчя та розрахували найкращу векторну систему для стиснення зображень. Потім Турк Майкл і Пентланд Алекс [13] застосували власні грані до проблеми розпізнавання осіб.

Аналіз головних компонентів – це метод проєкції на підпростір, який широко використовується в розпізнаванні образів. Метою PCA є заміна корельованих векторів великих розмірів некорельованими векторами менших розмірів [14]. Іншою метою є обчислення основи для набору даних. Основними перевагами PCA є низька чутливість до шуму, зниження вимог до пам'яті та ємності, а також підвищення ефективності за рахунок роботи в просторі менших розмірів.

Стратегія методу власних граней полягає у виділенні характерних рис обличчя та представленні відповідного обличчя як лінійної комбінації так званих «власних граней», отриманих у процесі виділення ознак.

Розраховано головні компоненти граней навчальної множини. Розпізнавання досягається за допомогою проєкції обличчя в простір, утворений власними гранями. Проводиться порівняння на основі евклідової відстані власних векторів власних граней і власної грані досліджуваного зображення. Якщо ця відстань досить мала, особу впізнають. З іншого боку, якщо відстань занадто велика, зображення вважається таким, що належить індивідууму, для якого систему потрібно навчити.

На рисунку 1.4 зображена блок-схема алгоритму. Як відправні точки зчитуються навчальні зображення розмірами $N \times N$ і вони перетворюються на розміри $N^2 \times 1$. Таким чином створюється навчальний набір розмірів $N^2 \times M$, де M є кількістю зразків зображень.

Середнє значення набору зображень обчислюється як:

$$\omega = \frac{1}{M}\sum_{i=1}^{M} G_i\omega = \frac{1}{M}\sum_{i=1}^{M} G_i, \tag{1.3}$$

де $\omega\omega$ – середнє зображення;

M – кількість зображень;

$G_i G_i$ – вектор зображення.

Рисунок 1.4 – Блок-схема алгоритму методу власних граней

Власні грані, що відповідають найвищим власним значенням, зберігаються. Ці власні грані визначають простір граней. Власний простір створюється проєктуванням зображення на простір граней, утворений власними гранями. Таким чином обчислюються вагові вектори. Розміри зображення налаштовуються відповідно до специфікацій, а зображення покращується на етапах попередньої обробки розпізнавання. Порівнюються вектор ваги зображення та вектори ваги облич у базі даних.

Середнє обличчя обчислюється та віднімається від кожного обличчя в навчальному наборі. За результатами операції віднімання формується матриця (A).

Різниця між кожним зображенням і середнім зображенням обчислюється як:

$$\gamma_i = G_i - \omega, i = 1,2, \dots M, \tag{1.4}$$

де γ_i – різниця між зображенням і середнім зображенням.

Матриця, отримана операцією віднімання (A), множиться на її транспонування і, отже, коваріаційна матриця C формується:

$$C = A^T * AC = A^T * A, \tag{1.5}$$

де A утворено різними векторами,

$$A = [\gamma_1, \gamma_2, \dots, \gamma_m]. \tag{1.6}$$

Розмірність матриці $C = N \times N$. Кількість M зображень використовуються для формування матриці C. На практиці розміри матриці $C = N \times M$. З іншого боку, оскільки $A = M$, лише M із N власних векторів є відмінними від нуля.

Розраховуються власні значення коваріаційної матриці. Власні грані створюються з використанням кількості навчальних зображень мінус кількість класів (загальна кількість людей) власних векторів.

Вибраний набір власних векторів множиться на матрицю A, щоб створити скорочений підпростір власних граней. Власні вектори менших власних значень відповідають меншим варіаціям коваріаційної матриці. Дискримінаційні риси обличчя зберігаються. Кількість власних векторів залежить від точності, з якою визначена база даних, і її можна оптимізувати. Групу вибраних власних векторів називають власними гранями. Після отримання власних граней зображення в базі даних проєктуються в простір власних граней, а ваги зображення в цьому просторі зберігаються. Щоб визначити ідентичність зображення, власні коефіцієнти порівнюються з власними коефіцієнтами в базі даних. Формується власне обличчя розглянутого зображення.

Розраховуються евклідові відстані між власною гранню зображення та власними гранями, збереженими раніше.

Особа, про яку йде мова, визначається як така, чия евклідова відстань є мінімальною, нижчою порогового значення в базі даних власних граней. Якщо

всі розраховані евклідові відстані більші за порогове значення, то зображення є невпізнанним.

Причини вибору методу власних граней для розпізнавання обличчя:

– його незалежність від геометрії обличчя;

– простота реалізації;

– можливість реалізації в реальному часі навіть без спеціального обладнання;

– легкість і швидкість розпізнавання порівняно з іншими методами;

– вищий рівень розпізнання в порівнянні з іншими методами.

Проблемою методу розпізнавання облич власних облич є час обчислення. Якщо база даних велика, це можливо знадобиться деякий час, щоб відновити особу особи, про яку йдеться.

Таким чином, перевагою методу власних облич є простота реалізації та незалежність від геометрії обличчя. Однак, недоліком є час порівняння облич, так як при великій базі даних час порівняння може бути великим.

1.3 Особливості визначення ступеню схожості облич

Технологія розпізнавання обличчя є скрізь: від розблокування смартфону до лову втікачів. Прогнози [15] показують, що це зростання використання та розвитку розпізнавання облич буде тільки зростати в майбутньому. Незважаючи на те, що багато компаній успішно користуються цими технологіями, вони, ймовірно, пройшли через різні перешкоди, щоб розробити або впровадити їх.

Досягнення високої точності є однією з головних проблем у розробці технологій розпізнання. Обличчя, яке представляється системі, може мати різні варіації світла, зміщення обличчя або зміна поз, певний тип прикусу тощо. Наприклад, у ситуації глобальної пандемії або в лабораторії, де потрібно

постійно носити улюблені маски, систему розпізнавання може бути важко реалізувати.

Системи розпізнавання існують деякий час, однак нещодавно піддався серйозній перевірці. Головним чином це пов’язано з етичними міркуваннями, які раптово усвідомлюють користувачі та розробники технології. Скептицизм головним чином пов’язаний з неефективністю та упередженнями [16] в моделях розпізнавання обличчя. Особливо в правоохоронній сфері дані системи вважаються дуже суперечливими. Багато випадків [17] неправомірного ув’язнення можна пояснити неточними та несправними системами розпізнавання обличчя. Ось чому багато компаній припинили використовувати, а інші вагаються з його впровадженням.

Системи розпізнавання створюють значну загрозу безпеці для користувачів, оскільки використовують біометричні дані (зображення обличчя), які можна легко використати для крадіжки особистих даних та інших зловмисних цілей. Наприклад, на щорічній конвенції хакерів Black Hat у Лас-Вегасі дослідникам вдалося зламати face-ID Apple протягом 120 с.

1.4 Аналіз літературних джерел щодо апробації результатів стосовно розпізнавання та визначення ступеню схожості облич

У статті [1] описано дослідження та оцінка бази FERET для вимірювання прогресу в розробці алгоритму та визначення майбутніх напрямків досліджень. Програма FERET мала намір створити велику базу даних зображень обличчя, зібрану незалежно від розробників алгоритму. Керувати базою даних було обрано доктора Гаррі Векслера з Університету Джорджа Мейсона. Збір бази даних був спільним зусиллям доктора Векслера та доктора Філліпса. Зображення були зібрані в напівконтрольованому середовищі. Щоб підтримувати певну узгодженість у всій базі даних, під час кожного фотосеансу використовувалися однакові фізичні налаштування. Оскільки для

кожного сеансу обладнання доводилося збирати заново, зображення, зібрані в різні дати, мали незначні відмінності.

У статті [2] описана система студентського аудиту за допомогою розпізнавання обличчя. Запропонована система використовує класифікатори Хаара, KNN, CNN, SVM, генеративні змагальні мережі та фільтри Габора. Після розпізнавання обличчя звіти про відвідування будуть створені та збережені у форматі Excel. Система тестується в різних умовах, таких як освітлення, рухи голови, зміна відстані між учнем і камерами. Після ретельного тестування розраховується загальна складність і точність. Запропонована система виявилася ефективним і надійним пристроєм для контролю відвідуваності в класі без витрат часу та ручної роботи. Розроблена система є економічно ефективною та вимагає легке встановлення.

Стаття [3] пропонує використовувати цю технологію спостереження для ідентифікації злочинців, які перебувають у втечі за їхніми попередніми злочинами. Цих злочинців можна впізнати за допомогою розпізнавання облич із зображення чи відеокадру, знятих камерами, встановленими в різних місцях, а також можна використовувати для ідентифікації зниклих дітей. Недоліком є те, що зображення зазвичай розмиті, мають меншу чіткість і не розпізнаються людським оком. Запропонована система може успішно розпізнавати більше одного обличчя, що корисно для швидкого пошуку підозрюваних осіб, оскільки час обчислення дуже малий. Створюється унікальний шаблон для кожного обличчя та порівнюється з іншими зображеннями, доступними в наборі даних. Якщо знайдено відповідність для введеного обличчя, то відображатимуться деталі, пов’язані з відповідним зображенням. Система зменшить злочинність і забезпечить безпеку суспільству.

Стаття [4] пропонує навчити нейрону мережу FaceNet за допомогою набору даних VGGFace2. Впровадження методу FaceNet у дослідження з використанням двох типів попередньо навчених моделей, а саме CASIA-WebFace і VGGFace2, і протестованих на різних наборах даних стандартних зображень обличчя, які широко використовувалися раніше. За результатами

цього дослідницького експерименту FaceNet показав чудові результати та перевершив інші методи. Використовуючи попередньо підготовлені моделі VGGFace2, FaceNet може досягти 100% точності на наборах даних YALE, JAFFE, AT & T, Essex faces95, Essex grimace, 99,375% для набору даних Essex faces94 і найгірших 77,67% для набору даних faces96.

У статті [5] подано огляд мережі DeepFace та її навчання на наборі даних LFW. Звичайна система глибокого розпізнавання обличчя включає в себе кілька основних компонентів: глибока мережа, функція оптимізації втрат, алгоритм класифікації та збір даних поїзда. Прагнучи забезпечити повне та всебічне дослідження таких складних структур, у цій статті спочатку обговорюється еволюція пов’язаних мережевих архітектур. Далі дається порівняльний аналіз функцій втрат, алгоритмів класифікації та наборів даних граней. Потім представлено порівняльне дослідження найсучасніших систем розпізнавання облич. Тут продуктивність систем обговорюється за допомогою трьох наборів даних порівняльного аналізу зі зростаючим ступенем складності. Крім того, було проведено експериментальне дослідження для порівняння кількох відкрито доступних фреймворків розпізнавання облич з точки зору точності та швидкості розпізнавання.

Стаття [6] пропонує переваги мережі ArcNet та показує її тренування на 10 тестах розпізнавання. Запропонований ArcFace має чітку геометричну інтерпретацію через точну відповідність геодезичній відстані на гіперсфері. Представлено найширшу експериментальну оцінку всіх останніх найсучасніших методів розпізнавання облич за десятьма тестами розпізнавання облич, яка включає нову великомасштабну базу даних зображень із трильйонами пар і великомасштабний набір відеоданих. Показано, що ArcFace стабільно перевершує сучасні технології та може бути легко реалізований із незначними обчислювальними витратами.

Стаття [7] пропонує використання мережі OpenFace для розпізнавання одного та кількох облич. Використовувався модуль розгортання Torch і Python

глибокого розпізнавання облич на основі нейронної мережі, який був точно визначений у часі.

У статті [8] описано використання VGGFace для розпізнавання на базі Python та TensorFlow 2.0. Використано keras-vggface і MTCNN, щоб допомогти створити модель Keras для VGGFace2 для розпізнавання облич. В публікації представлено код для створення системи відвідування.

Стаття [9] розповідає про сім популярних технологій з розпізнавання облич у 2023 році. Серед представлених технологій є використання розпізнавання в детекціях брехні та перевірці віку, оплаті в криптовалюті, персональний помічник, розширені можливості для штучного інтелекту, захист від хакерів, відстеження уваги водіїв для запобігання автомобільних аварій, розпізнавання облич під час COVID-19.

У статті [10] пропонується огляд роботи сіамських нейронних мереж, їх архітектури та основні способи застосування. Дві нейронні мережі є перцептронами прямого зв’язку та використовують зворотне поширення помилок під час навчання; мережі працюють паралельно в тандемі та порівнюють свої результати в кінці, як правило, через косинусну відстань. Вихід, згенерований виконанням сіамської нейронної мережі, можна вважати семантичною подібністю між прогнозованим представленням двох вхідних векторів. У цьому огляді спочатку описуємо архітектуру сіамської нейронної мережі, а потім окреслюємо її основні застосування в ряді обчислювальних областей з моменту її появи в 1994 році.

Стаття [11] пропонує тренування сіамської мережі з обмеженим вилученням (BESNet) та аналіз її контрастної функції. BESNet – це спільна навчальна мережа, в яку вбудовано новий модуль багатомасштабного виділення меж (MSBE). Таким чином, традиційні та глибокі методи навчання використовуються для спільного навчання, щоб максимізувати свої сильні сторони через співпрацю. Зокрема, для оптимізації BESNet використовується нова функція втрат з обмеженням вилучення (BEC) у поєднанні з функцією скорочувальних втрат. Враховуючи взаємодію між різними вилученими

функціями, розроблено стратегію об’єднання перетасування каналів, щоб використовувати додаткові переваги між функціями. Наші експерименти показують, що запропонований BESNet може значно покращити продуктивність CD та створити більш повні та чіткіші межі об’єктів. Експерименти, проведені на двох реальних наборах даних на різних сценах, демонструють його найсучаснішу продуктивність.

У статті [12] описано навчання моделі FaceNet на базі даних облич Labeled Faces in the Wild (LFW) та аналіз контрастних та триплетних втрат. Для навчання ми використовуємо трійки приблизно вирівняних відповідних/невідповідних патчів обличчя, згенерованих за допомогою нового онлайнового методу видобутку триплетів. Перевагою нашого підходу є набагато більша репрезентативна ефективність: досягнуто найсучаснішої продуктивності розпізнавання облич, використовуючи лише 128 байт на обличчі. На широко використовуваному наборі даних система LFW досягла нового рекорду точності 99,63%. На YouTube Faces DB він досягає 95,12%. Система знижує рівень помилок порівняно з найкращим опублікованим результатом на 30% для обох наборів даних.

У статті [13] розглядається підхід розпізнавання обличчя як проблему двовимірного розпізнавання, користуючись тим фактом, що обличчя зазвичай стоять вертикально, і тому їх можна описати невеликим набором двовимірних характерних зображень. Зображення облич проєктуються на простір ознак («простір обличчя»), який найкраще кодує варіації серед відомих зображень облич. Простір граней визначається «власними гранями», які є власними векторами набору граней; вони не обов’язково відповідають ізольованим ознакам, таким як очі, вуха та носи. Фреймворк надає можливість навчитися розпізнавати нові обличчя без нагляду.

У статті [14] пропонується створення системи аутентифікації користувача за допомогою розпізнавання обличчя на базі методу головних компонент. Система контролю за обличчям людини розробляється на основі поточної технології розпізнавання зображень, щоб виявляти обличчя

студентів у класі та позначати присутність людини, якщо вона або вона збігаються з даними обличчя в даній базі даних облич. Ця система HFR буде самостійно позначати присутність учнів у класі, не заважаючи вчителю, щоб учитель міг підсилити свою роль викладання різними способами.

Інтернет-ресурс [15] пропонує графік об’ємів ринку розпізнавання обличчя з 2019 по 2032 рік.

У джерелі [16] представлено можливі базові чинники (моделювання на основі даних і сценаріїв) і методологічні міркування для оцінки расового упередження в алгоритмах. Представлені фактори, керовані даними (наприклад, якість зображення, статистичні дані про популяцію зображень і архітектуру алгоритму), а також фактори моделювання сценаріїв, які враховують роль «користувача» алгоритму (наприклад, порогові рішення та демографічні обмеження). Щоб проілюструвати, як вирішуються ці проблеми, представлено дані чотирьох алгоритмів розпізнавання облич (алгоритм попереднього покоління та три глибокі згорткові нейронні мережі, DCNN) для облич східно-азіатських і кавказьких.

Інтернет-ресурс [17] розповідає про вразливість систем розпізнавання облич та як люди становляться раптовими правопорушниками. Описані випадки неправомірного обвинувачення людей у злочинах та проблемах використання мережі Clearview AI. Генеральний прокурор Нью-Джерсі Гурбір С. Грюол наклав мораторій на використання Clearview поліцією та оголосив про розслідування «цього продукту або подібних йому продуктів».

Таким чином, аналіз вище перерахованих джерел допомагає розібратися в методах та підходах для розпізнавання облич та визначення ступені їх схожості.

1.5 Постановка задачі дослідження

На сьогодення актуальність даної роботи є важливою, оскільки системи розпізнання та визначення ступеню схожості облич, незважаючи на наявні наукові статті та дослідження з даної тематики, не є добре розвинутими та інтегрованими в широкий спектр сфер життя людей. Треба знайти оптимальні методи та підходи для визначення ступеню схожості та вирішення пов’язаних з цією сферою проблем.

Об’єктом дослідження є розпізнавання та визначення ступеню схожості облич.

Метою дослідження є вивчення та порівняння методів для визначення ступеню схожості облич.

Враховуючи мету роботи, необхідні вирішити поставлені завдання:

– дослідити існуючі методи розпізнання та визначення ступеню схожості облич;

– проаналізувати особливості вихідних даних щодо визначення ступеню схожості облич;

– обрати методи та моделі для реалізації навчання з метою розпізнавання облич;

– виконати всі етапи розроблення застосунку для розпізнавання та визначення ступеню схожості облич;

– проаналізувати отримані результати;

– визначити перспективи подальшої роботи.

2 ОСОБЛИВОСТІ МЕТОДІВ РОЗПІЗНАВАННЯ ТА ВИЗНАЧЕННЯ СТУПЕНЮ СХОЖОСТІ ОБЛИЧ

2.1 Особливості методів розпізнавання облич

Виявлення та розпізнавання обличчя є однією з ключових систем автентифікації на основі біометрії, яка може використовуватися як для процесу автентифікації, так і для спостереження. З кожним днем стрімко зростає кількість шахрайств, розпізнавання облич стає для нас надзвичайно важливою системою. Різноманітні застосування ефективної системи розпізнавання обличчя включають криміналістику, ідентифікацію злочинців, спостереження, виключення шахрайства тощо.

Проведено багато досліджень як на національному, так і на міжнародному рівнях, але навіть після безперервних досліджень неможливо створити справді стійку та ефективну систему. які можуть добре працювати як у звичайних умовах, так і в умовах реального часу. Розпізнавання облич завжди було дуже складним завданням. Його справжній тест полягає в окресленні автоматизованої структури, яка відповідає людській здатності сприймати обличчя. Як би там не було, існує обмеження людських можливостей, коли вона керує великою кількістю незрозумілих видимостей.

Отже, потрібна запрограмована автоматична електронна структура з відносно вищою точністю розпізнавання та швидкою обробкою.

Процес виявлення та розпізнавання обличчя складається таких кроків (рис. 2.1):

Крок 1. Виявлення облич: для визначення облич на зображенні чи відео за допомогою орієнтирів на обличчі, як очі, ніс, рот тощо.

Крок 2. Виділення ознак: вирівнювання, нормалізація облич для кращого розпізнавання точність.

Крок 3. Розпізнавання обличчя: для розпізнавання конкретної людини на зображенні чи відео зіставлення з базою даних.

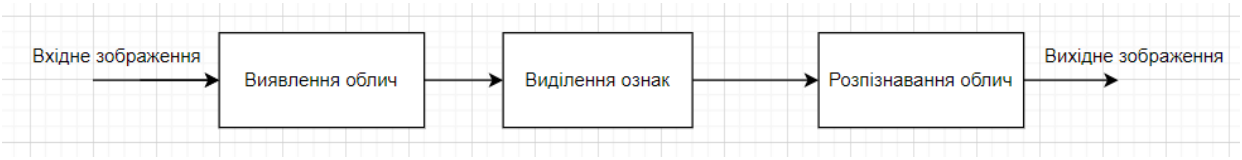


Рисунок 2.1 – Етапи виявлення та розпізнавання обличчя

Розпізнавання облич із зображень і відео – важке завдання. Проведено багато досліджень, щоб досягти 100% точності, але все ще не отримуємо задовільних результатів з огляду на різні чинники, що стикаються з цією системою.

Встановлено, що факторами, які знижують точність систем розпізнавання обличчя, є: оклюзія, низька роздільна здатність, шум, освітленість, зміна пози, вираз обличчя, старіння та пластична хірургія. Ці фактори можна класифікувати на дві категорії: внутрішні та зовнішні фактори.

Внутрішні компоненти включають фізичний стан людського обличчя, як старіння, вираз обличчя, пластична хірургія , впливаючи на систему, тоді як зовнішні фактори відповідальні за зміну зовнішнього вигляду обличчя, такі як оклюзія, низька роздільна здатність, шум, освітлення та зміна пози (рис. 2.2).

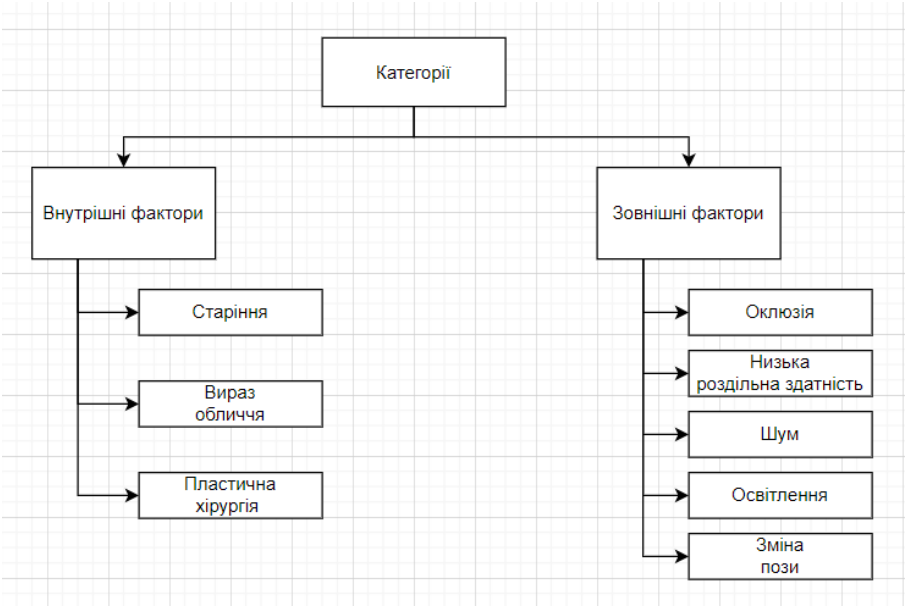


Рисунок 2.2 – Категорії факторів,
що впливають на точність розпізнавання обличчя

2.1.1 Оклюзія

Часткова оклюзія є однією з основних проблем розпізнавання обличчя. Було б важко розпізнати обличчя, якщо його частина відсутня. Наприклад, сонцезахисні окуляри та окуляри можуть приховати очі, сережки та волосся можуть приховати вуха, шарфи можуть приховати половину обличчя, вуса та борода чоловіків можуть приховати половину обличчя (рис. 2.3 [18]). Ці фактори можуть погіршити продуктивність системи. Для подолання цих проблем дослідники пропонують різні підходи [19].

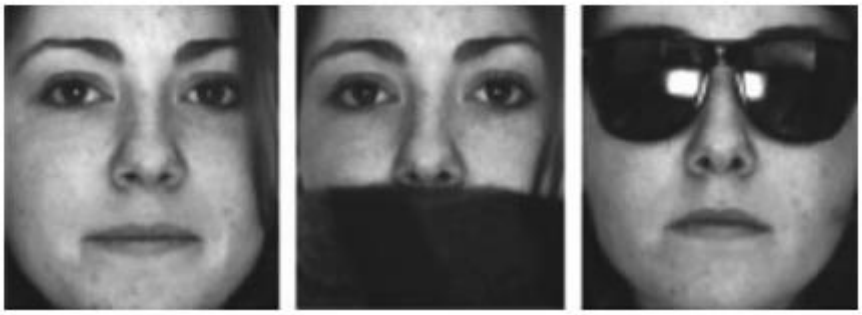
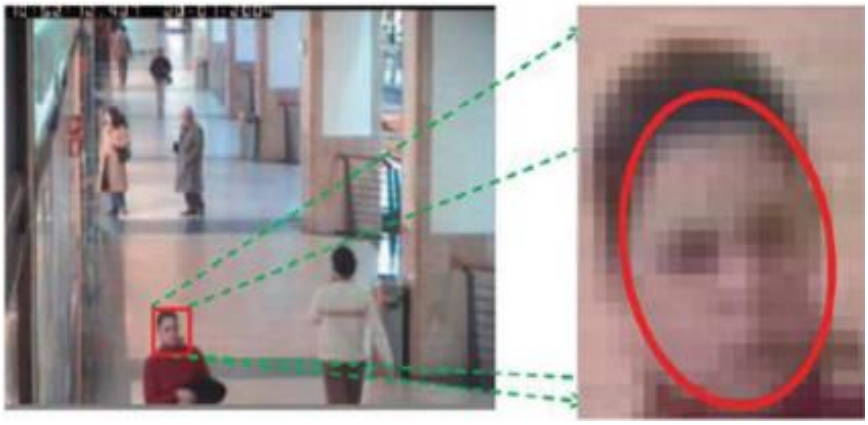


Рисунок 2.3 – Часткова оклюзія на обличчі

2.1.2 Низька роздільна здатність

На знімках з камер відеоспостереження зображені маленькі обличчя. Таким чином, його роздільна здатність низька, як показано на рисунку 2.4 а) та рисунку 2.4 б) [18].

Порівняти зображення з низькою роздільною здатністю та зображенням бази з високою роздільною здатністю є складним завданням. Таке зображення з низькою роздільною здатністю містить винятково обмежені дані, оскільки більша частина об’єктів для розпізнавання втрачається. Це може різко знизити швидкість розпізнавання. Для вирішення цієї проблеми дослідниками пропонуються окремі методології [19].



а) б)

Рисунок 2.4 – Кадр з відеоспостереження:
а) відео; б) захоплене обличчя

2.1.3 Шум

Цифрові зображення схильні до різного роду шумів. Цей шум призводить до низької точності виявлення та розпізнавання. Шум може бути введений у зображення різними способами, які залежать від створення зображення. Попередня обробка є обов’язковим фактором у загальній системі виявлення та розпізнавання облич [20]. На рисунку 2.5 а) зображено «чисте» зображення, на рисунку 2.5 б) показано шум «солі та перцю», присутній на зображенні [21].

а) б)

Рисунок 2.5 – Зображення: а) «чисте»; б) з шумом «сіль та перець»

2.1.4 Освітлення

Зміни в освітленні можуть різко погіршити продуктивність системи розпізнавання обличчя. Причинами цих варіацій можуть бути фонове освітлення, тінь, яскравість, контраст тощо. Зображення, зроблені в різних умовах освітлення, показані на рисунку 2.6 [18].

Рисунок 2.6 – Ефекти освітлення на зображеннях з обличчям

Різні підходи до освітлення обговорюються в [22].

2.1.5 Зміна пози

Розподіл різних поз також є однією з головних проблем системи розпізнавання обличчя. Реконструкція фронтального обличчя необхідна, щоб зіставити профільне обличчя з галерейним обличчям [23–29].

Ця реконструкція необхідна, оскільки зображення бази даних складається з фронтального вигляду та нефронтального профілю обличчя, що може давати хибні результати [30].

Дослідники пропонують різні підходи для перетворення нелобового обличчя на лобове обличчя могло підвищити точність розпізнавання [30].

Як зміна пози погіршує продуктивність алгоритму, обговорюється дослідниками в запропонованих підходах [31].

Різні розподіли ракурсів обличчя показані на рисунку 2.7 [18].

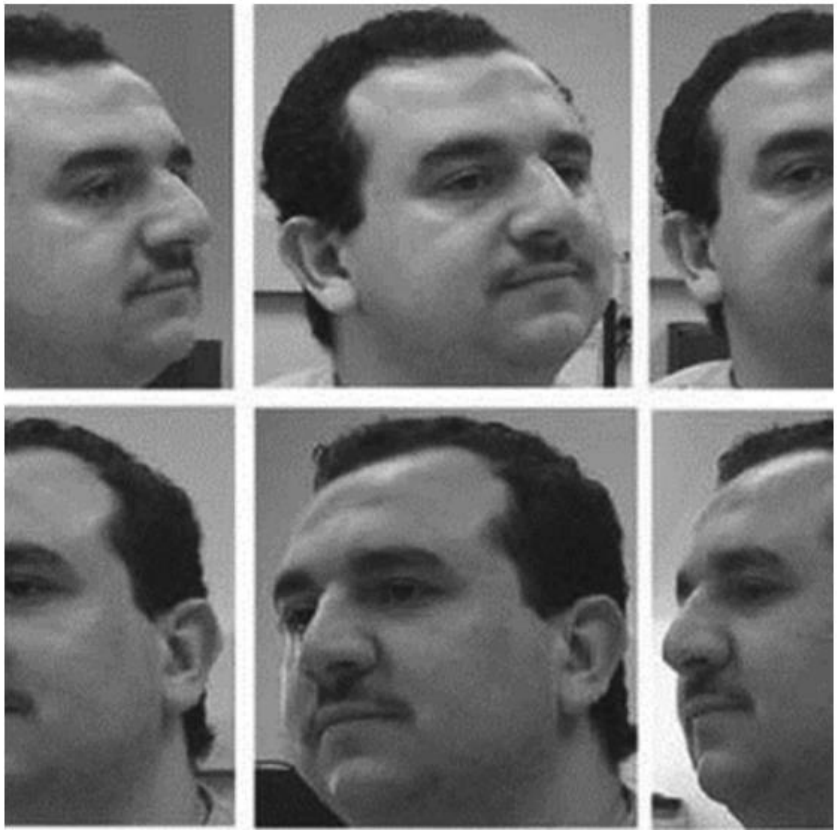


Рисунок 2.7 – Зміна пози

2.1.6 Вираз обличчя

За допомогою міміки можемо виражати свої почуття, як показано на рисунку 2.8 [18]. Це змінює геометрію обличчя. Зміна обличчя може створити нечіткість для системи розпізнавання обличчя. Вираз обличчя – це швидкий сигнал, на який легко впливати через скорочення м’язів і змінює риси обличчя, наприклад, брови, щоки, рот тощо. Постійно проводяться дослідження для розпізнавання обличчя з урахуванням виразу обличчя [32].

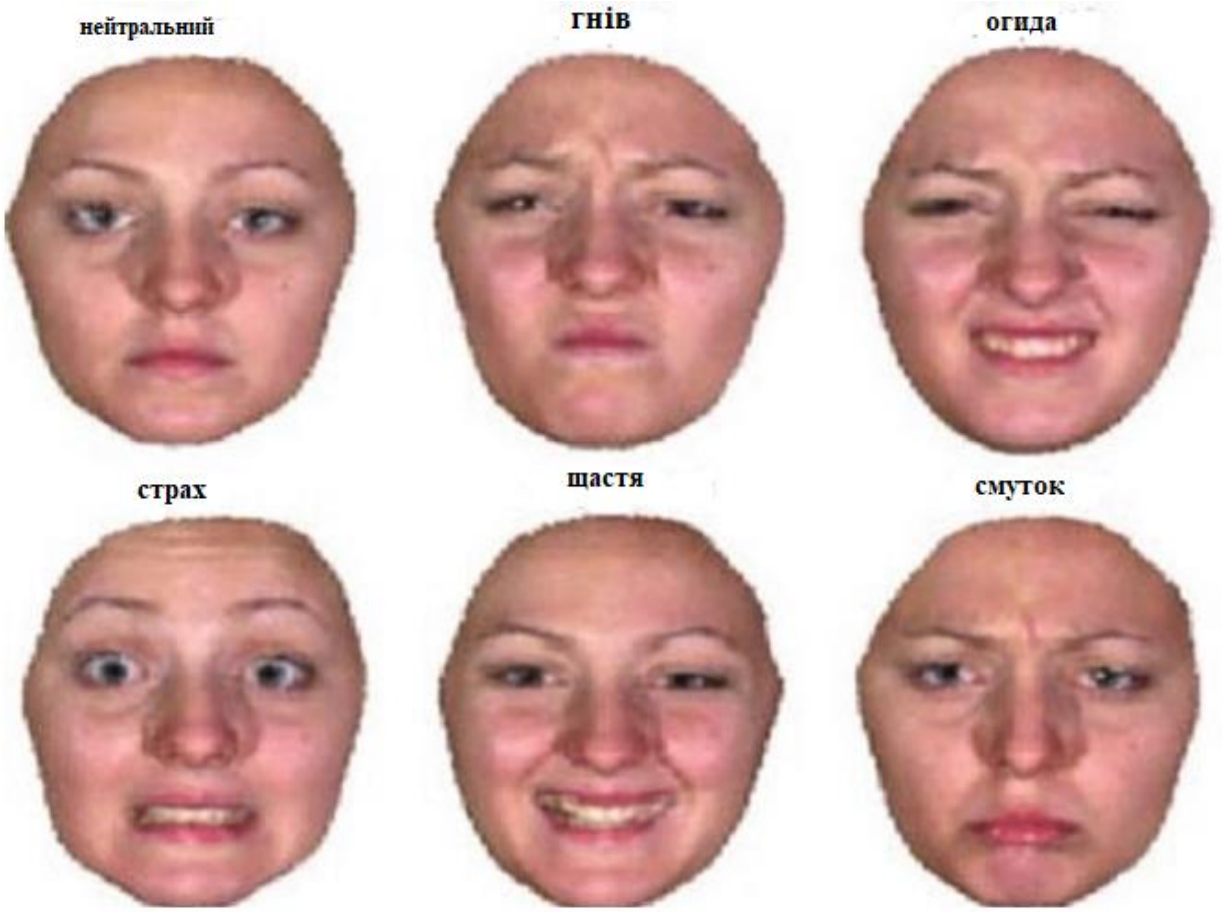


Рисунок 2.8 – Різні вирази обличчя

2.1.7 Старіння

Старіння є одним із природних компонентів, що впливає на системи розпізнавання облич, оскільки воно стає крахом для алгоритму.

Обличчя – це суміш тканин шкіри, м’язів обличчя та кісток. У той момент, коли м’язи скорочуються, вони призводять до скручування відблисків обличчя. Роки викликають критичні зміни в зовнішності особи, наприклад, поверхні обличчя (зморшки тощо) і формі обличчя з плином часу. Фреймворки розпізнавання облич повинні бути достатньо кваліфікованими для врахування цієї вимоги [33].

Різна фактура обличчя однієї людини в різному віці наведена на рисунку 2.9 [18].

Рисунок 2.9 – Вікові варіації людини

2.1.8 Пластична хірургія

Результати пластичної хірургії – це також головний фактор, який впливає на точність розпізнавання обличчя. Є багато випадків, коли через нещасні випадки багато людей зазнали пластичних операцій, і їхні обличчя будуть невідомі існуючій системі розпізнавання облич. Здебільшого злочинці приймають ідею пластичної хірургії, щоб приховати свою особистість.

Отже, як обговорювалося в [34], потрібна система ідентифікації, яка здатна розпізнавати обличчя навіть після реконструктивної хірургії. Ефект пластичної хірургії показаний на рисунку 2.10 [34].

а) б)

Рисунок 2.10 – Наслідки хірургії: а) до; б) після

Отже, є багато особливостей, які стають на перепоні ефективного розпізнаванні облич, серед яких пластична хірургія та вік особи.

2.2 Механізм визначення ступеню схожості облич

Модель FaceNet, запропонована Шроффом та іншими, вирішує проблему перевірки обличчя. Він приймає зображення обличчя партіями та навчається на них за допомогою функції триплетних втрат для розрахунку втрат. Пакет містить зображення як позитивні, негативні та опорні пари. Під час обчислення втрат функція мінімізує відстань між прив’язкою та позитивом, тобто зображеннями однієї особистості, і максимізує відстань між прив’язкою та негативом, тобто зображеннями різних ідентичностей. Він вивчає один глибокий CNN, а потім перетворює зображення обличчя на вбудовування. Вбудовування можна використовувати для порівняння облич трьома способами:

– перевірка обличчя розглядає два обличчя та вирішує, схожі вони чи ні. Перевірку обличчя можна виконати шляхом обчислення метрики відстані;

– розпізнавання облич є проблемою класифікації для позначення обличчя іменем. Вектор вбудовування можна використовувати для навчання остаточних міток;

– розпізнавання облич є проблемою класифікації для позначення обличчя іменем. Вектор вбудовування можна використовувати для навчання остаточних міток.

Вивчення подібності – це процес навчання математичної функції або метрики для вимірювання ступеня зв’язку між елементами. Він просто вимірює показник обох спостережуваних елементів, також відомий як навчання метриці. Таким чином, векторні вбудовування, згенеровані з моделі FaceNet, можна порівняти одне з одним, щоб перевірити подібність.

Традиційні алгоритми для вимірювання подібності можуть бути такими:

– метриками подібності на основі відстані є евклідова відстань (2.1), манхеттенська відстань (2.2), відстань Мінковського (2.3). Основна ідея, спільна для цих показників, полягає в тому, що вони використовують середню відстань між елементами двох експериментальних векторів.

$$d(p,q) = \sqrt{(p1-q1)^2 + (p2-q2)^2 + \cdots + (pn-qn)^2}, \qquad (2.1)$$

$$d(p,q) = |px-qx| + |py-qy| + \cdots + |pn-qn|, \qquad (2.2)$$

$$d(p,q) = \left(\sum_{i=1}^{n} |pi-qi|^p\right)^{\frac{1}{p}}; \qquad (2.3)$$

– метрики подібності на основі кардинальності використовують об’єднання та перетин множин у порівнянні. Метрика подібності Жаккара використовує потужність для визначення відношення між двома експериментальними векторами

$$d(p,q) = (\frac{p \cap q}{p \cup q}); \quad (2.4)$$

– метрики подібності на основі орієнтації використовують кут між двома векторами у відповідних векторних просторах. Прикладом є метрика косинусної подібності, яка обчислює подібність шляхом вимірювання косинуса кута між двома векторами

$$d(p,q) = \cos(\theta) = \frac{\sum_{i=1}^{n} pi \times qi}{\sqrt{\sum_{i=1}^{n} pi^2} \times \sqrt{\sum_{i=1}^{n} qi^2}}. \quad (2.5)$$

2.3 Методика розпізнавання та визначення ступеню схожості облич

Вилучення ознак – це метод визначення набору ознак, який показує представлення інформації, необхідної для аналізу та класифікація зображень у різні класи в розпізнаванні образів [35]. Основна проблема розпізнавання обличчя полягає в тому, як отримати інформацію із зображень обличчя або фотографій. Виділення ознак можна розглядати як процедуру отримання важливої інформації із зображення обличчя. Ця інформація має бути цінною для наступного етапу ідентифікації суб'єкта зі стандартною частотою помилок.

Процес вилучення ознак має бути ефективним з точки зору часу обчислення та використання пам'яті, вихід також має бути оптимізовано для класифікації крок.

Різні методи виділення ознак можна згрупувати у дві основні категорії: підхід на основі зовнішнього вигляду (цілісний) і підхід на основі ознак (геометричний).

2.3.1 Методи на основі зовнішнього вигляду

У підході на основі зовнішнього вигляду вся область обличчя береться до уваги як вхідні дані зображення для системи. Метод обробляє зображення обличчя як двовимірні шаблони, концепція риси в цьому підході відрізняється від простих рис обличчя, таких як очі та рот. Будь-яка витягнута характеристика із зображення відноситься до функції. Техніка найкраще виділяє риси обличчя оскільки він зберігає важливу інформацію зображення та відкидає зайву інформацію.

Існує кілька методів, що базуються на зовнішньому вигляді, які включають: аналіз основних компонентів, лінійний дискримінантний аналіз, проєкція зі збереженням лінійності, незалежний компонентний аналіз і лінійний дискримінантний аналіз [35].

Лінійний дискримінантний аналіз (LDA). Цей прийом також відомий як fisherface [35]. LDA є потужною технікою для зменшення даних і вилучення ознак, яка використовується для розробки системи розпізнавання обличчя, створює ефективне представлення, яке лінійно перетворює вихідний простір даних у низьковимірну функцію з фокусом на більшості дискримінантних ознак (виконайте розмірність). скорочення при збереженні якомога більшої кількості дискримінаційної інформації за класом) [36]. Техніка дає посилання на вектори ознак у базовому просторі, які найкраще представляють найкращі дискримінаційні ознаки серед класів, а не найкраще описують дані. LDA проєктує зображення обличчя з високовимірного простору зображення в низьковимірне зображення шляхом обчислювальної трансформації, яка максимізує розкид між класами, мінімізуючи в межах класу.

Проєкція зі збереженням лінійності (LPP) – це підхід виділення ознак, у якому головним чином спрямовано на поєднання переваг лінійних методів і локальних нелінійних методів зменшення розмірності шляхом знаходження лінійного відображення, яке мінімізує функцію вартості лапласівських власних карт [37].

LPP представляє лінійну апроксимацію нелінійних лапласівських власних відображень, коли високовимірні дані лежать на низькій розмірній різновиді, вбудованій у простір даних. Техніка зберігає локальну структуру даних, на відміну від PCA та LDA, які зберігають глобальну структуру ознак або даних.

PCA використовується для виділення ознак і представлення даних у комп'ютерному зорі та розпізнаванні образів, наприклад, розпізнавання обличчя [28]. Він зазвичай використовується для зменшення кількості рис обличчя для розпізнавання обличчя. Він шукає набір репрезентативних векторів ознак проєкції, наприклад, щоб спроєктовані зразки зберігали більшість інформації про оригінальні зразки.Метод PCA застосовує перетворення векторного простору, щоб зменшити розмірність великої бази даних. Застосування математичної проєкції, яка перетворює ряд можливо корельованих змінних на меншу кількість некорельованих змінних, які називаються головними компонентами.

2.3.2 Методи на основі ознак

Метод, заснований на ознаках, робить основний акцент на рисах обличчя, таких як очі, ніс і рот, а також на інших опорних знаках для побудови моделі на основі положення та розміру цих характеристик для формування вектора ознак. Потім застосовуються стандартні статистичні методи розпізнавання образів, щоб зіставити обличчя за допомогою цих вимірювань. Підходи за технікою на основі ознак включають: локальний бінарний шаблон. Вектор ознак представляє геометричні співвідношення обличчя між точками обличчя, таким чином зменшуючи вхідне зображення обличчя до вектора геометричні особливості [38]. Визначається відстань між елементами, яка використовується для представлення зображення обличчя. Це найпростіший і ранній метод для системи розпізнавання обличчя.

Проблема з цією технікою полягає в тому, що вона не може ідентифікувати обличчя з різним освітленням і точкою зору, а також ненадійна щодо часу.

Локальні бінарні шаблони (LBP) є іншим підходом до функції заснований на розпізнаванні обличчя, що використовується в класифікації в цифровій обробці зображень і комп'ютерному зорі. LBP є конкретною демонстрацією моделі Texture Spectrum, запропонованої в 1990 році. LBP вперше було виражено в 1994 році та справило великий вплив на біометричне поле. Було встановлено, що це впливовий підхід для класифікації текстури ознак. Було проаналізовано, що коли LBP поєднується з дескриптором гістограми, розробленим під час обчислень, це покращує ефективність виявлення на регулярній основі та точно на деяких наборах даних [39].

LBP призначений для опису текстур. Оператор призначається кожному пікселю зображення через порогове значення околиці 3×3 кожного пікселя відносно значення центрального пікселя та відповідного результату у вигляді двійкового числа. У цьому алгоритмі зображення обличчя ділиться на локальні області, і дескриптор текстури витягується з цих областей, і вони об'єднуються, щоб сформувати глобальний опис обличчя.

Використовуючи глобальний опис шаблонів, формується просторово покращена гістограма, яка кодує як просторові, так і зовнішні відносини зображення обличчя. Просторова розширена гістограма далі використовується для вимірювання відстані. Кожен елемент гістограми пов'язаний з невеликою ділянкою обличчя на основі психофізичних даних. Деякі риси обличчя відіграють важливу роль для розпізнавання обличчя. Зважена відстань χ^2 використовується для вимірювання відстані.

Рисунок 2.11 [40] описує маску 3×3, що містить значення. Визначається порогове значення та порівнюється з ним значення сусідніх пікселів. Для порівняння, двійковий код є розроблений і використовується для процесу розпізнавання.

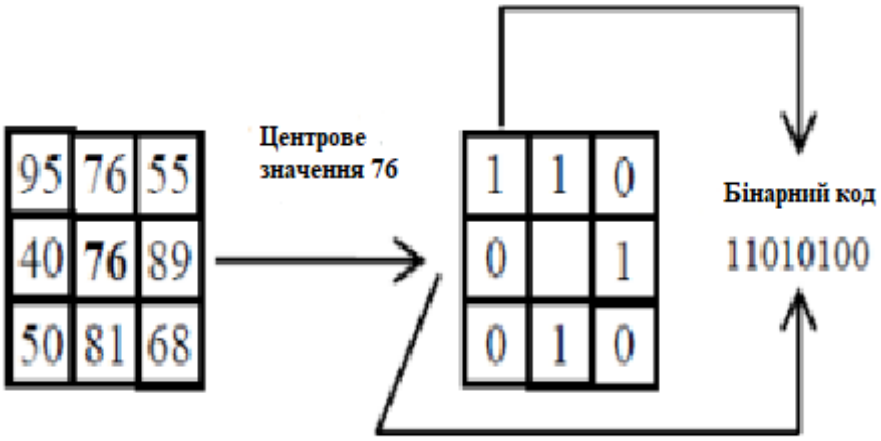


Рисунок 2.11 – Декодинг лінійних бінарних шаблонів

LBP вектор ознак найпростішим способом створюється таким чином:

– розділіть досліджуване вікно на клітинки (наприклад, 16×16 пікселів для кожної комірки);

– для кожного пікселя в комірці порівняйте піксель з кожним із 8 його сусідів (ліворуч угорі, ліворуч посередині, ліворуч унизу, праворуч угорі). Проведіть пікселі по колу, тобто за або проти годинникової стрілки;

– де значення центрального пікселя більше, ніж значення сусіда, напишіть «1». В іншому випадку напишіть «0». Це призводить до 8-значного двійкового числа (яке зазвичай перетворено в десяткове для зручності);

– обчисліть гістограму по комірці частоти кожного «числа», що зустрічається (тобто кожної комбінації які пікселі менші, а які більші за центр);

– додатково нормалізуйте гістограму;

– зчеплені (нормалізовані) гістограми всіх клітин і в результаті розробляється вектор ознак для вікна.

Цей підхід використовувався в багатьох різних варіантах програми, які включають завдання, пов’язані з виявленням обличчя, розпізнаванням обличчя, демографічною класифікацією та аналізом виразу обличчя [41].

3 ДОСЛІДЖЕННЯ МЕТОДІВ РОЗПІЗНАВАННЯ ТА ВИЗНАЧЕННЯ СТУПЕНЮ СХОЖОСТІ ОБЛИЧ

3.1 Вибір інструментальних засобів для реалізації методів розпізнавання та визначення ступеню схожості облич

У рамках кваліфікаційної роботи був розроблений алгоритм для розпізнавання зображень та порівняння їх ступеню схожості за допомогою нейронних мереж. Для реалізації було обране середовище Python 3.8. Для розпізнавання зображень була використана бібліотека DeepFace.

DeepFace – це проєкт із відкритим вихідним кодом, написаний мовою Python і ліцензований згідно з ліцензією MIT. Розробникам дозволяється використовувати, змінювати та поширювати бібліотеку як у приватному, так і в комерційному контексті.

DeepFace – найпростіша бібліотека розпізнавання облич і аналізу атрибутів обличчя для Python. Бібліотека DeepFace із відкритим вихідним кодом включає всі передові моделі штучного інтелекту для розпізнавання облич і автоматично виконує всі процедури розпізнавання облич у фоновому режимі.Бібліотека дозволяє отримати доступ до набору функцій:

– перевірка обличчя: завдання перевірки обличчя стосується порівняння обличчя з іншим, щоб перевірити, чи збігається воно чи ні. Тому перевірка обличчя зазвичай використовується для порівняння обличчя кандидата з іншим. Це можна використовувати для підтвердження того, що фізичне обличчя збігається з обличчям у документі, що посвідчує особу;

– розпізнавання обличчя: завдання стосується пошуку обличчя в базі даних зображень. Виконання розпізнавання обличчя вимагає запуску перевірки обличчя багато разів;

– аналіз атрибутів обличчя. Завдання аналізу атрибутів обличчя стосується опису візуальних властивостей зображень обличчя. Відповідно, аналіз атрибутів обличчя використовується для вилучення таких атрибутів, як

вік, класифікація статі, аналіз емоцій або прогнозування расової/етнічної приналежності;

– аналіз обличчя в реальному часі: функція включає тестування розпізнавання обличчя та аналіз атрибутів обличчя за допомогою відео в режимі реального часу вашої вебкамери.

У той час як більшість альтернативних бібліотек розпізнавання облич обслуговують одну модель штучного інтелекту, бібліотека DeepFace об'єднує багато передових моделей розпізнавання облич. Це найпростіший спосіб використовувати алгоритм Facebook DeepFace та всі інші найпопулярніші алгоритми розпізнавання обличчя.

З бібліотекою DeepFace можна використовувати наступні алгоритми глибокого навчання розпізнавання обличчя. Більшість із них базуються на найсучасніших згорткових нейронних мережах (CNN) і забезпечують найкращі результати в своєму класі:

– VGG-Face. VGG означає Visual Geometry Group. Нейронна мережа VGG (VGGNet) є одним із найбільш використовуваних типів моделей розпізнавання зображень на основі глибоких згорткових нейронних мереж. Архітектура VGG стала відомою завдяки досягненню найкращих результатів на конкурсі ImageNet. Модель розроблена дослідниками з Оксфордського університету. Хоча VGG-Face має ту саму структуру, що й звичайна модель VGG, він налаштований із зображеннями обличчя. Модель розпізнавання обличчя VGG досягає 97,78% точності на популярному наборі даних Labeled Faces in the Wild (LFW);

– Google Facenet. Ця модель розроблена дослідниками Google. FaceNet вважається найсучаснішою моделлю для виявлення та розпізнавання облич із глибоким навчанням. FaceNet можна використовувати для розпізнавання облич, перевірки та кластеризації (кластеризація облич використовується для кластеризації фотографій людей з однаковою ідентичністю). Основною перевагою FaceNet є його висока ефективність і продуктивність.

Повідомляється, що він досягає 99,63% точності в наборі даних LFW і 95,12% в Youtube Faces DataBase, використовуючи лише 128 байт на обличчя;

– OpenFace. Цю модель розпізнавання облич створили дослідники з Університету Карнегі-Меллона. Отже, OpenFace значною мірою натхненний проєктом FaceNet, але він більш легкий, а тип його ліцензії є більш гнучким. OpenFace досягає 93,80% точності на наборі даних LFW;

– Facebook DeepFace. Цю модель розпізнавання облич розробили дослідники Facebook. Алгоритм Facebook DeepFace був навчений на позначеному наборі даних із чотирьох мільйонів облич, що належать понад 4000 осіб, що було найбільшим набором даних про обличчя на момент випуску. Підхід базується на глибокій нейронній мережі з дев'яти рівнів. Модель Facebook досягає точності 97,35% (+/- 0,25%) за тестом набору даних LFW. Дослідники стверджують , що алгоритм DeepFace Facebook скоротить розрив у продуктивності на рівні людини (97,53%) на тому ж наборі даних. Це вказує на те, що DeepFace іноді успішніше, ніж люди, виконуючи завдання розпізнавання обличчя;

– ArcFace. Це найновіша модель в модельному ряду. Його спільними розробниками є дослідники Імперського коледжу Лондона та InsightFace. Модель ArcFace досягає 99,40% точності на наборі даних LFW.

Виявлення обличчя та вирівнювання є дуже важливими етапами конвеєра розпізнавання обличчя. Google заявив, що лише вирівнювання обличчя підвищує точність розпізнавання обличчя на 0,76%. Загалом, DeepFace – це простий спосіб використання найпопулярніших сучасних детекторів обличчя. Наразі в DeepFace включено кілька передових детекторів обличчя:

– OpenCV. Порівняно з іншими, OpenCV є найлегшим детектором обличчя [42, 43]. Популярний інструмент обробки зображень використовує каскадний алгоритм Хаара, який не базується на методах глибокого навчання. Тому він швидкий, але його продуктивність відносно низька. Для належної роботи OpenCV потрібні фронтальні зображення. Крім того, його

продуктивність виявлення очей середня. Це спричиняє проблеми з вирівнюванням. Зауважте, що детектором за замовчуванням у DeepFace є OpenCV;

– Dlib. Цей детектор використовує алгоритм hog у фоновому режимі. Отже, як і OpenCV, він не базується на глибокому навчанні. Тим не менш, він має відносно високі показники виявлення та вирівнювання;

– SSD. SSD означає Single-Shot Detector; це популярний детектор на основі глибокого навчання. Продуктивність SSD порівнянна з OpenCV. Однак SSD не підтримує орієнтири обличчя та залежить від модуля виявлення очей OpenCV для вирівнювання [44]. Незважаючи на високу ефективність виявлення, оцінка вирівнювання лише середня;

– MTCNN. Це детектор обличчя, заснований на глибокому навчанні, і він має орієнтири обличчя. Ось чому для MTCNN високі показники виявлення та вирівнювання. Однак він повільніший, ніж OpenCV, SSD і Dlib;

– RetinaFace. Це детектор обличчя, заснований на глибокому навчанні, і він має орієнтири обличчя. Ось чому для MTCNN високі показники виявлення та вирівнювання. Однак він повільніший, ніж OpenCV, SSD і Dlib.

3.2 Етапи програмної реалізації методів розпізнавання та визначення ступеню схожості облич

Розробка програмної реалізації методів розпізнавання та визначення ступеню схожості облич – це комплексний підхід, який складається з чотирьох основних етапів, кожний з яких має важливе значення для успішної реалізації системи розпізнавання та визначення ступеню схожості облич.

Перший етап – розпізнавання обличчя. На цьому етапі алгоритм намагається знайти зображення обличчя на зображенні. Для цього DeepFace містить портфоліо з чотирьох передових детекторів. Це OpenCV, SSD, RetinaFace і MTCNN. Найпотужнішими моделями є RetinaFace або MTCNN

для великої точності та високої впевненості в розпізнаванні. Але оскільки вони дійсно повільні, слід розглянути можливість використання OpenCV або SSD, коли швидкість важливіша. Оскільки алгоритм очікує зображення однакового розміру, DeepFace автоматично додає кілька чорних пікселів до кожного з них, щоб уникнути деформації.

Другий етап – це вирівнювання та нормалізація. Правильне вирівнювання підвищує точність розпізнавання обличчя більш ніж на один відсоток. Хоча це може звучати не дуже багато, уже є моделі з точністю людського рівня, тож можна змінити ситуацію навіть за невеликих ознак прогресу, як цей. Дуже корисно, що сучасні детектори мають здатність розпізнавати орієнтири обличчя, наприклад очі. Тому DeepFace виявляє їх і повертає зображення, доки координати очей не стануть горизонтальною лінією. Оскільки обличчя виявляються в прямокутній області, там також може зберігатися деяка інформація про шум. За допомогою нормалізації можна позбутися цього й просто зберегти інформацію про обличчя.

Третій етап є представленням зображення для використання згорткових нейронних мереж для класифікації зображення та порівняння. Вони відповідають за представлення зображень обличчя у вигляді векторів. CNN спочатку навчається класифікувати ідентичності, а потім початкові рівні витягають необроблений вектор ознак представлення обличчя. Таким чином перевіряється обличчя, яких раніше не бачив.

Четвертий етап є підтвердженням порівняння облич. Зображення обличчя представлено як вектори на попередньому етапі та очікувано, що векторні вбудовування однієї людини мають бути більш схожими на зображення іншої людини. Цю відстань можна виразити за допомогою формули евклідової відстані та відстані козіна. На основі оголошеного порогу алгоритм може вибрати, чи є відстань ближчою до «тієї самої людини» чи «іншої людини».

3.2.1 Розпізнавання обличчя

Не дивлячись на те, що методів для детектування обличчя багато, докладно розібравшись можна вибрати певний детектор під певні задачі. Виявлення обличчя є важливим етапом, щоб система розпізнавання облич була надійною.

Модель SSD в основному базується на моделі VGG – Face . Модель SSD передбачає, що будуть подані вхідні дані розміром (300, 300, 3). Змінимо розмір вхідних зображень до 300×300, але це низька роздільна здатність, і щоб не втратити роздільну здатність, також зберігаємо базове зображення в змінній.

Лістинг 3.1 Завантаження зображення та його обробка:

```
image = cv2.imread("image.jpg")
base_img = image.copy()
original_size = base_img.shape
target_size = (300, 300)
image = cv2.resize(image, target_size)
aspect_ratio_x = (original_size[1] / target_size[1])
aspect_ratio_y = (original_size[0] / target_size[0])
```

Це буде зображення, яке обробимо (рис. 3.1).

Вже змінили розмір базового зображення до 300×300 пікселів, але OpenCV насправді очікує, що буде подано вхідні дані (1, 3, 300, 300) точного розміру. Простіший спосіб – використовувати функцію blobFromImage. Якщо закодувати цю логіку, тоді можна альтернативно згорнути 3-й вимір до 1-го, а потім додати фіктивний вимір ліворуч, викликавши функцію розширення розмірів.

Рисунок 3.1 – Вихідне зображення

Лістинг 3.2 Зміна розміру зображення:

```
#detector expects (1, 3, 300, 300) shaped input
imageBlob = cv2.dnn.blobFromImage(image = image)
#imageBlob = np.expand_dims(np.rollaxis(image, 2, 0), axis = 0)
```

Вихід нейронних мереж є матрицею розміру (200, 7). Тут рядки стосуються облич кандидатів, а стовпці – деякі характеристики. Відфільтруємо ці обличчя кандидатів на основі їх характеристик (рис. 3.2).

Лістинг 3.3 Фільтрація облич:

```
column_labels = ["img_id", "is_face", "confidence", "left", "top", "right", "bottom"]
detections_df = pd.DataFrame(detections[0][0], columns = column_labels)
```

	img_id	is_face	confidence	left	top	right	bottom
0	0.0	1.0	0.999969	0.496539	0.138750	0.659196	0.338830
1	0.0	1.0	0.993426	0.239311	0.266163	0.395453	0.470608
2	0.0	1.0	0.347227	0.620609	0.450788	0.733679	0.544840
3	0.0	1.0	0.129311	4.156785	3.998088	4.839249	4.980163
4	0.0	1.0	0.121853	0.607522	0.641886	0.636585	0.677165

Рисунок 3.2 – Вивід даних по обличчям

Функція is_face матиме значення 0 для фону та 1 для екземплярів обличчя. Тому будемо ігнорувати нульові значення. Також будемо ігнорувати випадки, які мають значення довіри, нижче за порогове значення (наприклад, 90%).

Лістинг 3.4 Функція is_face:

```
#0: background, 1: face
detections_df = detections_df[detections_df['is_face'] == 1]
detections_df =
detections_df[detections_df['confidence']&amp;amp;amp;amp;lt;=0.90]
```

Значення координат – ліворуч, праворуч, зверху, знизу – знаходяться в [0, 1]. Розмір вхідного зображення змінено на (300, 300). Слід помножити ці значення координат на 300, щоб знайти точне розташування на зміненому зображенні.

Листинг 3.5 Зміна координат обличчя:

```
detections_df['left'] = (detections_df['left'] * 300).astype(int)
detections_df['bottom'] = (detections_df['bottom'] * 300).astype(int)
detections_df['right'] = (detections_df['right'] * 300).astype(int)
detections_df['top'] = (detections_df['top'] * 300).astype(int)
```

Застосування цих фільтрів усуває більшість екземплярів, і наступні два екземпляри з’являються у фреймі даних pandas (рис. 3.3).

	img_id	is_face	confidence	left	top	right	bottom
0	0.0	1.0	0.999969	148	41	197	101
1	0.0	1.0	0.993426	71	79	118	141

Рисунок 3.3 – Знайдені обличчя

Можна витягти виявлену область обличчя з базового зображення (рис. 3.4).

Лістинг 3.6 Витягування обличчя з зображення:

```
for i, instance in detections_df.iterrows():
confidence_score = str(round(100*instance["confidence"], 2))+" %"
left = instance["left"]; right = instance["right"]
bottom = instance["bottom"]; top = instance["top"]
detected_face =
base_img[int(top*aspect_ratio_y):int(bottom*aspect_ratio_y) ,
int(left*aspect_ratio_x):int(right*aspect_ratio_x)]
print("Id ",i,". Confidence: ", confidence_score)
plt.imshow(detected_face[:,:,::-1])
plt.show()
```

Рисунок 3.4 – Виявлені обличчя

Також можна зосередитися на виявлених обличчях на базовому зображенні (рис. 3.5).

Лістинг 3.7 Втягування облич з базового зображення:

*cv2.putText(base_img, confidence_score, (int(left*aspect_ratio_x), int(top*aspect_ratio_y-10)), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 1)*

*cv2.rectangle(base_img, (int(left*aspect_ratio_x), int(top*aspect_ratio_y)), (int(right*aspect_ratio_x), int(bottom*aspect_ratio_y)), (255, 255, 255), 1) #draw rectangle to main image*

Рисунок 3.5 – Виявлені обличчя на базовому рисунку 3.1

Модель RetinaFace. Дизайн моделі базується на пірамідах функцій. Контекстні модулі йдуть після незалежних контекстних модулів (рис. 3.6).

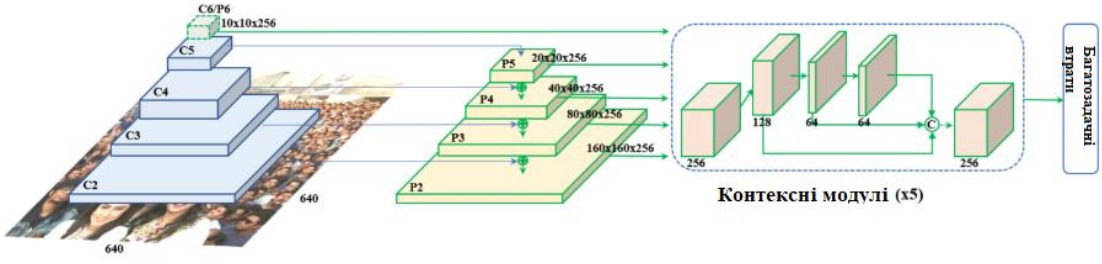


Рисунок 3.6 – Архітектура моделі RetinaFace

Піраміди функцій обробляються у вихідному коді, як показано нижче.

Лістинг 3.8 Ознаки піраміди RetinaFace:

```
#feature pyramids
model = Model(inputs=data,
outputs=[
  face_rpn_cls_prob_reshape_stride32,
  face_rpn_bbox_pred_stride32,
  face_rpn_landmark_pred_stride32,
  face_rpn_cls_prob_reshape_stride16,
  face_rpn_bbox_pred_stride16,
  face_rpn_landmark_pred_stride16,
  face_rpn_cls_prob_reshape_stride8,
  face_rpn_bbox_pred_stride8,
  face_rpn_landmark_pred_stride8
])
```

Після встановлення пакета можна імпортувати бібліотеку. У інтерфейсі є функція виявлення обличчя. Функція очікує точного шляху зображення. Передача зображень у форматі numpy також підходить.

Лістинг 3.9 Імпорт бібліотеки RetinaFace:

```
from retinaface import RetinaFace
img_path = "img1.jpg"
faces = RetinaFace.detect_faces(img_path)
```

Потім функція поверне координати області обличчя, деякі орієнтири, включаючи координати очей, носа та рота з оцінкою достовірності.

Лістинг 3.10 Координати області обличчя:

```
{
   "face_1": {
      "score": 0.9993440508842468,
      "facial_area": [155, 81, 434, 443],
      "landmarks": {
        "right_eye": [257.82974, 209.64787],
        "left_eye": [374.93427, 251.78687],
        "nose": [303.4773, 299.91144],
        "mouth_right": [228.37329, 338.73193],
        "mouth_left": [320.21982, 374.58798]
      }
  }
}
```


Можна виділити область обличчя та орієнтири, якщо у вас є функція виявлення облич (рис. 3.7).

Лістинг 3.11 Виділення області облич та орієнтири:

```
identity = faces[0]
facial_area = identity["facial_area"]
landmarks = identity["landmarks"]
#highlight facial area
cv2.rectangle(img, (facial_area[2], facial_area[3])
   , (facial_area[0], facial_area[1]), (255, 255, 255), 1)
#extract facial area
#img = cv2.imread(img_path)
#facial_img = img[facial_area[1]: facial_area[3], facial_area[0]: facial_area[2]]
#highlight the landmarks
 cv2.circle(img, tuple(landmarks["left_eye"]), 1, (0, 0, 255), -1)
```

cv2.circle(img, tuple(landmarks["right_eye"]), 1, (0, 0, 255), -1)

cv2.circle(img, tuple(landmarks["nose"]), 1, (0, 0, 255), -1)

cv2.circle(img, tuple(landmarks["mouth_left"]), 1, (0, 0, 255), -1)

cv2.circle(img, tuple(landmarks["mouth_right"]), 1, (0, 0, 255), -1)

Рисунок 3.7 – Розпізнання облич RetinaFace

Модель MTCNN.MTCNN є потужним детектором обличчя з високими показниками виявлення. Це розшифровується як багатозадачна каскадна згорткова мережа.

MTCNN базується на 3 окремих моделях CNN: P-Net, R-Net і O-Net (рис. 3.8).

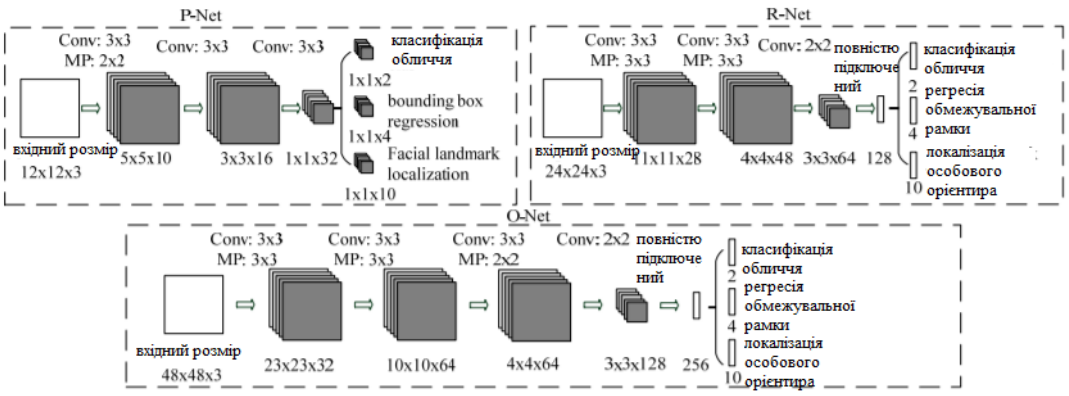


Рисунок 3.8 – Архітектура MTCNN: P-Net, R-Net, O-Net

Назва P-Net походить від мережі пропозицій. Шукає обличчя в рамках розміром 12×12. Місія цієї мережі – отримати швидкі результати.

Назва R-Net походить від refine network.

Мережа має більш глибоку структуру, ніж P-Net. Усі кандидати, які прийшли з попередньої мережі P-Net, передаються в R-Net. P-Net тут відхиляє величезну кількість кандидатів. Нарешті, вихідна мережа або незабаром O-Net повертає обмежувальну рамку (область обличчя) і положення орієнтирів обличчя.

MTCNN – це максимально легке рішення. Спочатку створимо детектор MTCNN і подамо масив numpy як вхідні дані для функції виявлення облич під його інтерфейсом. Завантажимо вхідне зображення за допомогою OpenCV у наступний блок коду. Функція виявлення облич повертає масив об’єктів для виявлених облич. Повернений об’єкт зберігає координати виявлених облич у ключі поля.

Лістинг 3.12 Розпізнавання обличчя за допомогою MTCNN:

```
from mtcnn import MTCNN
import cv2
detector = MTCNN()
img = cv2.imread("img.jpg")
detections = detector.detect_faces(img)
for detection in detections:
score = detection["confidence"]
if score &amp;amp;gt; 0.90:
x, y, w, h = detection["box"]
detected_face = img[int(y):int(y+h), int(x):int(x+w)]
```

Незважаючи на те, що модель SSD на основі OpenCV забезпечує такий самий рівень точності, MTCNN також знаходить деякі орієнтири на обличчі, наприклад розташування очей, носа та рота.

Зокрема, виділення розташування очей є дуже важливим для вирівнювання облич. Вирівнювання обличчя підвищує точність моделі розпізнавання обличчя майже на 1% згідно з дослідженням Google FaceNet.

З іншого боку, OpenCV знаходить розташування очей за допомогою традиційного каскадного методу Хаара, який є недостатньо ефективним. Іншими словами, потрібно покладатися на застарілий каскад Хаара в OpenCV, щоб вирівняти грані, навіть якщо використовується сучасний SSD.

Повернений об’єкт функції виявлення облич також зберігає орієнтири обличчя.

Лістинг 3.13 Орієнтири розташування очей:

```
keypoints = detection["keypoints"]
left_eye = keypoints["left_eye"]
right_eye = keypoints["right_eye"]
```

3.2.2 Вирівнювання та нормалізація

Вирівнювання обличчя є ранньою стадією сучасного розпізнавання обличчя. Google заявив, що вирівнювання обличчя підвищує точність його моделі розпізнавання обличчя FaceNet з 98,87% до 99,63%. Це майже 1% підвищення точності. Подібно до виявлення обличчя, яке також є попереднім етапом конвеєра, можемо легко застосувати 2D вирівнювання обличчя в OpenCV у Python.

Зосередимося на виявленому обличчі та ігноруватимемо поза зоною (рис. 3.9).

Лістинг 3.14 Виявлення обличчя:

```
faces = face_detector.detectMultiScale(img, 1.3, 5)
face_x, face_y, face_w, face_h = faces[0]
```

```
img = img[int(face_y):int(face_y+face_h), int(face_x):int(face_x+face_w)]
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

Рисунок 3.9 – Виявлене обличчя

OpenCV також пропонує виявлення очей. Він очікує сіру версію зображення (рис. 3.10).

Лістинг 3.15 Виявлення очей:

```
eyes = eye_detector.detectMultiScale(gray_img)
index = 0
for (eye_x, eye_y, eye_w, eye_h) in eyes:
if index == 0:
eye_1 = (eye_x, eye_y, eye_w, eye_h)
elif index == 1:
eye_2 = (eye_x, eye_y, eye_w, eye_h)
cv2.rectangle(img,(eye_x, eye_y),(eye_x+eye_w, eye_y+eye_h), color, 2)
index = index + 1
```

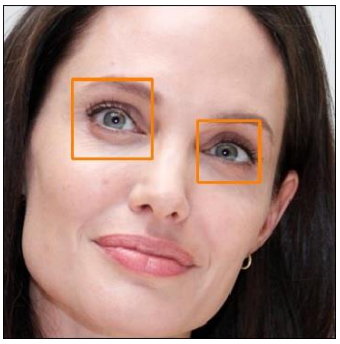


Рисунок 3.10 – Виявлення очей

Збережемо розташування очей у змінних 1-го та 2-го ока. Тепер треба визначити, яке око ліве, а яке праве. Розташування *X* зберігаються в 1-му елементі кортежу. Отже, маленький квадрат буде лівим оком.

Лістинг 3.16 Визначення розташування очей:

```
if eye_1[0] &amp;amp;amp;amp;lt; eye_2[0]:
left_eye = eye_1
right_eye = eye_2
else:
left_eye = eye_2
right_eye = eye_1
```

Координати ока. Верхня ліва точка – це точка (0, 0) у OpenCV. Виявлене око містить 4 значення. На наступній ілюстрації показано ці значення (рис. 3.11).

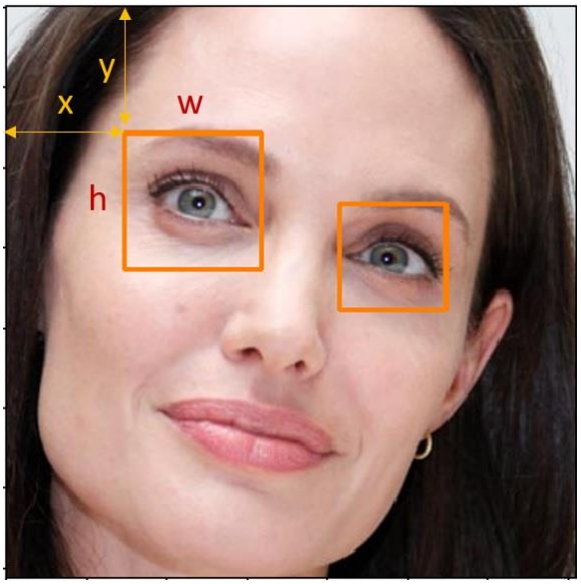


Рисунок 3.11 – Координати очей

Потрібен центр виявлених очей. Індекс 0 відноситься до *x*, індекс 1 відноситься до *y*, індекс 2 відноситься до *w* і індекс 3 відноситься до *h*. Давайте також проведемо лінію між центром двох очей (рис. 3.12).

Лістинг 3.17 Координати очей:

left_eye_center = (int(left_eye[0] + (left_eye[2] / 2)), int(left_eye[1] + (left_eye[3] / 2)))

left_eye_x = left_eye_center[0]; left_eye_y = left_eye_center[1]

right_eye_center = (int(right_eye[0] + (right_eye[2]/2)), int(right_eye[1] + (right_eye[3]/2)))

right_eye_x = right_eye_center[0]; right_eye_y = right_eye_center[1]

cv2.circle(img, left_eye_center, 2, (255, 0, 0) , 2)

cv2.circle(img, right_eye_center, 2, (255, 0, 0) , 2)

cv2.line(img,right_eye_center, left_eye_center,(67,67,67),2)

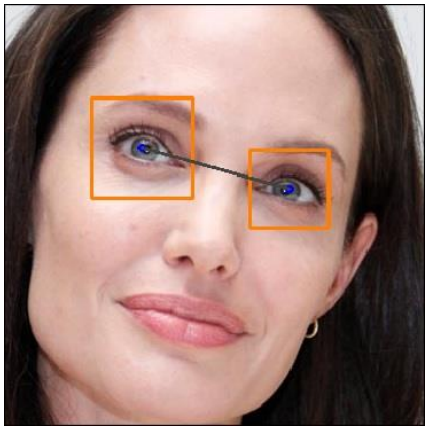


Рисунок 3.12 – Лінія поєднання очей

Потрібен кут між горизонтальною лінією і намальованою лінією. Оскільки будемо обертати зображення на основі цього кута. На рисунку 3.13 ліве око вище правого. Тому будемо обертати зображення проти напрямку годинника. З іншого боку, повернули б зображення відповідно до напрямку годинника, якби праве око було вище, ніж праве око.

Лістинг 3.18 Кут між очима:

if left_eye_y < right_eye_y:

point_3rd = (right_eye_x, left_eye_y)

```
direction = -1 #rotate same direction to clock
print("rotate to clock direction")
else:
point_3rd = (left_eye_x, right_eye_y)
direction = 1 #rotate inverse direction of clock
print("rotate to inverse clock direction")
cv2.circle(img, point_3rd, 2, (255, 0, 0) , 2)
cv2.line(img,right_eye_center, left_eye_center,(67,67,67),2)
cv2.line(img,left_eye_center, point_3rd,(67,67,67),2)
cv2.line(img,right_eye_center, point_3rd,(67,67,67),2)
```

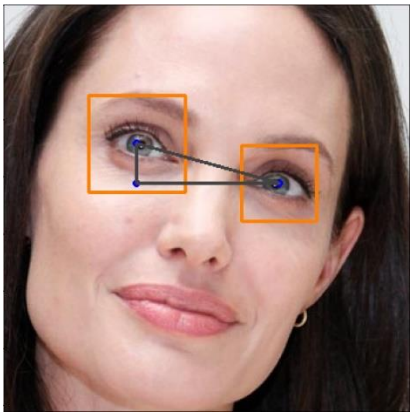


Рисунок 3.13 – Кут між очима

Можна знайти довжину 3 ребер трикутника з евклідовою відстанню (рис. 3.14).

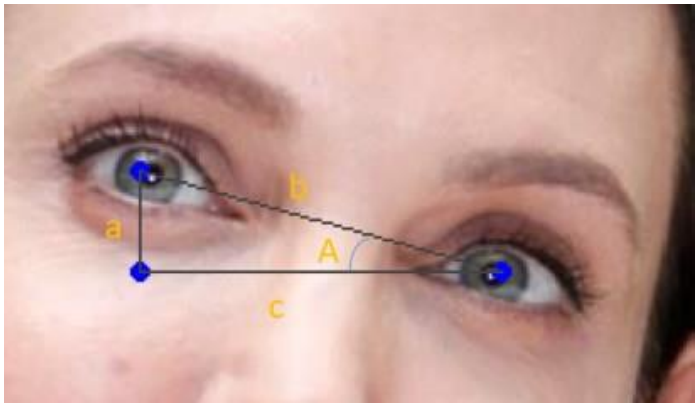


Рисунок 3.14 – Правило косинуса

Лістинг 3.19 Правило косинуса:

```
def euclidean_distance(a, b):
x1 = a[0]; y1 = a[1]
x2 = b[0]; y2 = b[1]
return math.sqrt(((x2 - x1) * (x2 - x1)) + ((y2 - y1) * (y2 - y1)))
a = euclidean_distance(left_eye_center, point_3rd)
b = euclidean_distance(right_eye_center, left_eye_center)
c = euclidean_distance(right_eye_center, point_3rd)
```

Після обчислення довжини ребер трикутника можна застосувати правило косинуса. Знаходження кута за його значенням косинуса потребує виклику обернених тригонометричних функцій. Однак функція arccos повертає кут у радіанах. Потрібно знайти 180-кратне значення Пі, щоб знайти значення в градусах.

Лістинг 3.20 Визначення кута нахилу:

```
cos_a = (b*b + c*c - a*a)/(2*b*c)
print("cos(a) = ", cos_a)
angle = np.arccos(cos_a)
print("angle: ", angle," in radian")
angle = (angle * 180) / math.pi
print("angle: ", angle," in degree")
if direction == -1:
angle = 90 - angle
```

Іншими словами, один кут дорівнює 90 градусам. Сума інших двох кутів також буде дорівнювати 90 градусам. Якщо будемо обертати зображення в напрямку годинника, то потрібно повернути зображення на 90 мінус

знайдений кут. Можна зрозуміти чому, подивившись на наступну демонстрацію (рис. 3.15).

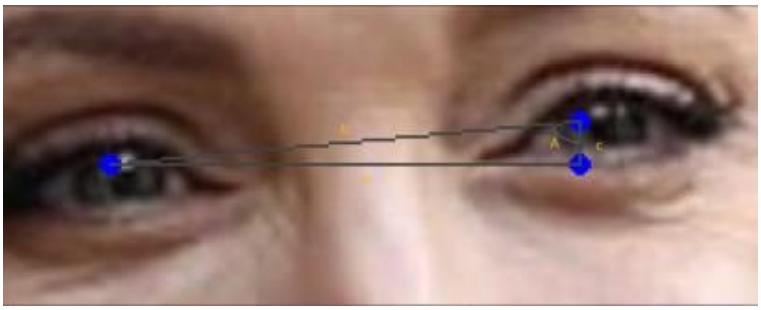


Рисунок 3.15 – Обертання в напрямку годинника

3.2.3 Представлення зображення

Представлення зображення для використання згорткових нейронних мереж для класифікації зображення та порівняння. Вони відповідають за представлення зображень обличчя у вигляді векторів. CNN спочатку навчається класифікувати ідентичності, а потім початкові рівні витягають необроблений вектор ознак представлення обличчя. Таким чином перевіряється обличчя, яких раніше не бачив.

Таблиця 3.1 показує форму вхідних даних тих моделей розпізнавання обличчя, які підтримуються в нашій системі. Вихідні фігури виражають кількість розмірів, у яких буде представлено вхідне зображення.

Таблиця 3.1 – Розміри вхідних та вихідних даних моделей

Модель	**Вхідні розміри**	**Вихідний розмір**
VGG-Face	224×224×3	2622
FaceNet	160×160×3	128
OpenFace	96×96×3	128
DeepFace	152×152×3	4096
ArcFace	112×112×3	512

Модель VGG-Face має спільну структуру як imagenet (рис. 3.16). VGG очікує вхідних зображень розміром 224×224×3. Таким чином можемо представити зображення як 2622-вимірний вектор, як показано нижче.

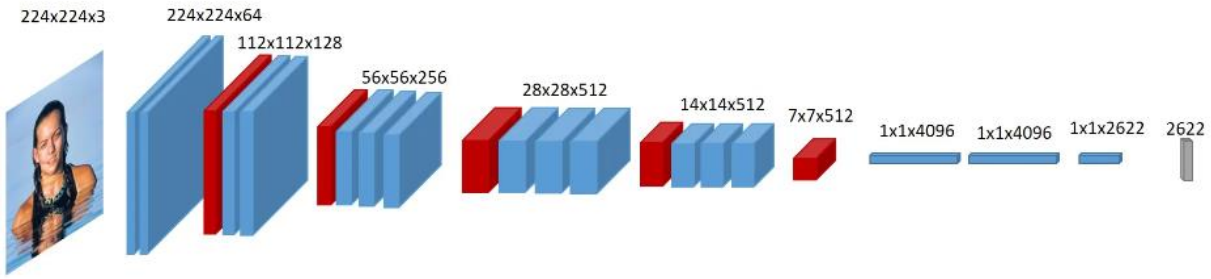


Рисунок 3.16 – Архітектура VGG-Face

Модель FaceNet очікує зображення розміром 160×160, тоді як вона створює 128-вимірні зображення.

Модель OpenFace очікує зображення (96×96) RGB як вхідні дані. Вона має 128-вимірний вихід. Модель побудована на Inception Resnet V1.

Модель ArcFace в основному базується на моделі ResNet34. Модель ArcFace очікує (112, 112, 3) вхідних даних у формі, тоді як вона повертає 512 вимірних векторних зображень.

3.2.4 Підтвердження порівняння облич

Етап подання представляє зображення обличчя як вектори. Модуль подання передає два однакові вектори розмірності в модуль перевірки. Розглянемо подібності цих двох векторів у цьому шарі. Щоб знайти відстані між векторами можна обчислити косинусну подібність, евклідову відстань або її форму L2.

Багатовимірний вектор є дуже абстрактним поняттям, оскільки не можна візуалізувати його в тривимірному просторі. Могли б візуалізувати багатовимірний вектор, щоб зробити його конкретним. На рисунку 3.17

показано представлення VGG-Face двох зображень. VGG-Face представляє зображення у вигляді 2622-вимірних векторів. На панелі праворуч від зображення є 2622 слоти. Кожен слот представляє вимір у вихідному векторі. Його значення також визначає колір слота.

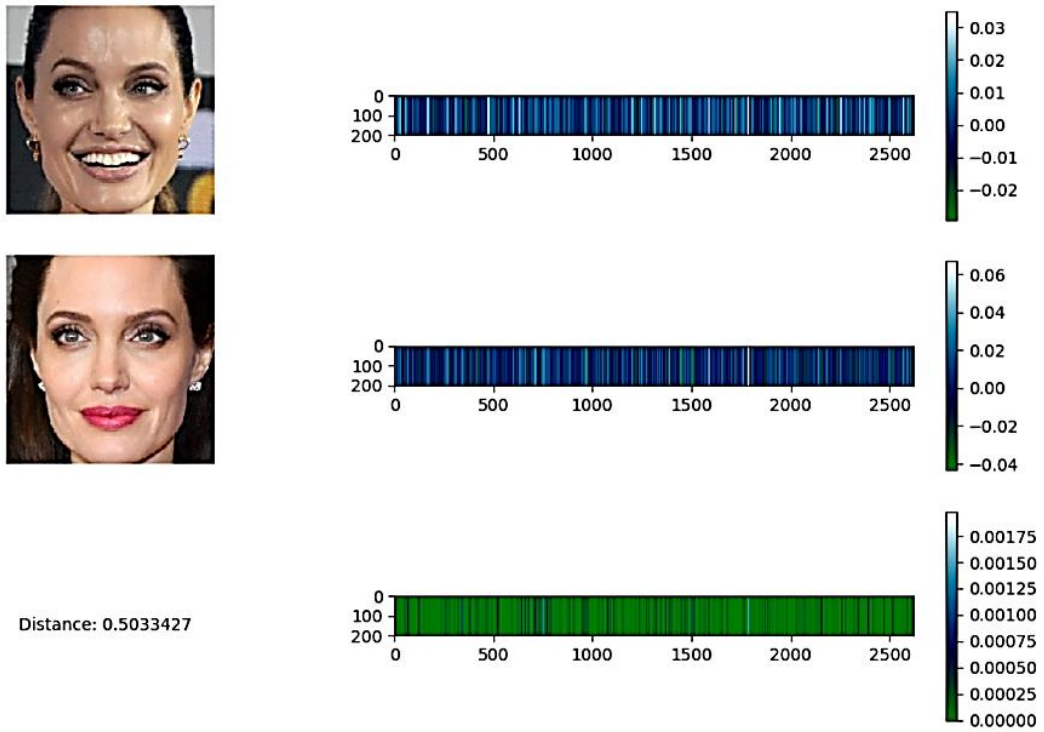


Рисунок 3.17 – Візуалізація евклідової відстані

Евклідова відстань вимагає знайти квадрат різниці кожного виміру. Тут потрібно знайти різницю кожного слота та знайти квадратне значення. Гістограма в 3-му рядку зберігає ці квадратичні значення різниці. Потрібно підсумувати всі слоти та знайти значення квадратного кореня, щоб знайти евклідову відстань.

Косинусну подібність двох векторів виводять за допомогою формули евклідового скалярного добутку, зазначеної у формулі (3.1). Тут *x* і *y* різні багатовимірні вектори. Значення подібності знаходиться в діапазоні [-1, +1]. Подібні вектори повинні мати велике значення подібності, тоді як дуже різні вектори повинні мати мале значення подібності. І навпаки, подібні вектори повинні мати невелике значення евклідової відстані. Ось чому можна

перетворити його на косинусну відстань, згадану у формулі (3.2), яка дорівнює 1 мінус косинус подібності. Таким чином, малість і величина виражають одне і те ж як у косинусі, так і в евклідовій відстані.

$$\cos\theta = \frac{\sum_{i=0}^{n} x_i \times y_i}{\sqrt{\sum_{i=0}^{n} x_i^2}\sqrt{\sum_{i=0}^{n}\sum_{i=0}^{n} y_i^2}}, \quad (3.1)$$

$$D_c(x, y) = 1 - \cos\theta. \quad (3.2)$$

Нижче приведено реалізація евклідової та косинусної відстаней в коді.

Лістинг 3.21 Евклідова та косинусна відстані:

```
def findCosineDistance(source_representation, test_representation):
a = np.matmul(np.transpose(source_representation), test_representation)
b = np.sum(np.multiply(source_representation, source_representation))
c = np.sum(np.multiply(test_representation, test_representation))
return 1 - (a / (np.sqrt(b) * np.sqrt(c)))
def l2_normalize(x, axis=-1, epsilon=1e-10):
output = x / np.sqrt(np.maximum(np.sum(np.square(x), axis=axis, keepdims=True), epsilon))
return output
def findEuclideanDistance(source_representation, test_representation):
euclidean_distance = source_representation - test_representation
euclidean_distance = np.sum(np.multiply(euclidean_distance, euclidean_distance))
euclidean_distance = np.sqrt(euclidean_distance)
#euclidean_distance = l2_normalize(euclidean_distance )
return euclidean_distance
cosine = findCosineDistance(img1_representation, img2_representation)
```

euclidean = findEuclideanDistance(img1_representation, img2_representation)

Поріг. У нас є значення відстані, але як визначити відстань мала чи висока? Найпростіше це визначити, нажививши багато позитивних і негативних екземплярів. Тоді алгоритми дерева рішень можуть знайти найкращу точку розділення. Для кожного методу та формули подібності є свої пороги для якісного порівняння облич.

У даному підрозділі було надано опис програмної реалізації методів розпізнавання та визначення ступеню схожості облич. Розглянуто процес нормалізації зображення,верифікації за допомогою евклідової та косинусної формули.

3.3 Тестування розроблених застосунків та аналіз результатів

Тестування застосунку для порівняння схожості облич потребує комплексного підходу та велику кількість перевірок моделей на парі зображень. Для порівняння облич використовуються три метрики: згорткова нейронна мережа, детектор облич та формула схожості. За порівняння візьмемо два обличчя Анджеліни Джолі (рис. 3.18 та рис. 3.19).

Рисунок 3.18 – Анджеліна Джолі

Рисунок 3.19 – Анджеліна Джолі

Дві картинки будемо порівнювати в парах: нейронна мережа, детектор облич та формули подібності.

За результатами таблиць 3.2 – 3.5 можна сказати, що найшвидша та найрезультативніша зв’язка це VGG-Face та MNCNN.MTCNN, де видно точний результат за менший час завдяки своїй архітектурі. Поганим результатом вийшла зв’язка з детектором SSD. Всі результати були вище порогового значення.

Таблиця 3.2 – Результати VGG-Face та OpenCV

VGG-Face та OpenCV	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Так	0,25	0,4	7,85
euclidean	Так	0,49	0,6	4,27
euclidean_l2	Так	0,71	0,86	3,5

Таблиця 3.3 – Результати VGG-Face та RetinaFace

VGG-Face та Retinaface	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Так	0,24	0,4	6,9
euclidean	Так	0,43	0,6	6,44
euclidean_l2	Так	0,70	0,86	6,4

Таблиця 3.4 – Результати VGG-Face та MTCNN

VGG-Face та MTCNN	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Так	0,27	0,4	4,36
euclidean	Так	0,46	0,6	4,13
euclidean_l2	Так	0,73	0,86	4,01

Таблиця 3.5 – Результати VGG-Face та SSD

VGG-Face та SSD	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Ні	0,46	0,4	8,33
euclidean	Ні	0,601	0,6	3,24
euclidean_l2	Ні	0,96	0,86	3,97

За результатами таблиць 3.6 – 3.9 можна сказати, що найрезультативніша зв’язка це DeepFace та MNCNN. Хоча всі детектори показали результат за майже однаковий час, MTCNN показує точний результат за менший час завдяки своїй архітектурі. Поганим результатом вийшла зв’язка з детектором SSD. Всі результати були вище порогового значення.

Таблиця 3.6 – Результати DeepFace та OpenCV

DeepFace та OpenCV	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Ні	0,24	0,23	28,03
euclidean	Так	63	64	26,27
euclidean_l2	Ні	0,69	0,64	27,21

Таблиця 3.7 – Результати DeepFace та RetinaFace

DeepFace та Retinaface	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Так	0,16	0,23	29,85
euclidean	Так	58	64	29,18
euclidean_l2	Так	0,58	0,64	29,29

Таблиця 3.8 – Результати DeepFace та MTCNN

DeepFace та MTCNN	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Так	0,14	0,23	27,25
euclidean	Так	56	64	26,72
euclidean_l2	Так	0,54	0,64	26,67

За результатами таблиць 3.10 – 3.13 можна сказати, що найкраща зв’язка за часом та результатом це FaceNet та OpenCV. Всі результати мали такий час завдяки малій кількості параметрів у мережі, а це 22,7 млн. OpenCV показує точний результат за менший час завдяки своїй архітектурі. Поганим результатом вийшла зв’язка з детектором SSD. Всі результати були вище порогового значення, хоча за часом була сама найшвидша.

Таблиця 3.9 – Результати DeepFace та SSD

DeepFace та SSD	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Ні	0,37	0,23	26,54
euclidean	Ні	89	64	25,94
euclidean_l2	Ні	0,86	0,64	26,13

Таблиця 3.10 – Результати FaceNet та OpenCV

FaceNet та OpenCV	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Так	0,24	0,4	4,86
euclidean	Так	8,14	10	4,63
euclidean_l2	Так	0,69	0,8	4,64

Таблиця 3.11 – Результати FaceNet та RetinaFace

FaceNet та RetinaFace	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Так	0,21	0,4	7,82
euclidean	Так	7,50	10	7,7
euclidean_l2	Так	0,65	0,8	7,68

Таблиця 3.12 – Результати FaceNet та MTCNN

FaceNet та MTCNN	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Так	0,20	0,4	5,31
euclidean	Так	7,5	10	5,63
euclidean_l2	Так	0,64	0,8	5,28

Таблиця 3.13 – Результати FaceNet та SSD

FaceNet та SSD	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Ні	0,57	0,4	4,36
euclidean	Ні	11,23	10	4,78
euclidean_l2	Ні	1,07	0,8	4,38

За результатами таблиць 3.14 – 3.17 можна сказати, що найрезультативніша зв’язка це OpenFace та MNCNN. MTCNN показує точний результат за менший час завдяки своїй архітектурі. Поганим результатом вийшли зв’язки з детекторами SSD та OpenCV. Всі результати були вище порогового значення.

Таблиця 3.14 – Результати OpenFace та OpenCV

OpenFace та OpenCV	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Ні	0,15	0,1	3,28
euclidean	Ні	0,56	0,55	3,19
euclidean_l2	Ні	0,56	0,55	3,22

Таблиця 3.15 – Результати OpenFace та RetinaFace

OpenFace та RetinaFace	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Так	0,08	0,1	6,15
euclidean	Так	0,42	0,55	6,17
euclidean_l2	Так	0,42	0,55	6,27

Таблиця 3.16 – Результати OpenFace та MTCNN

OpenFace та MTCNN	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Ні	0,13	0,1	3,76
euclidean	Так	0,51	0,55	3,67
euclidean_l2	Так	0,51	0,55	3,91

Таблиця 3.17 – Результати OpenFace та SSD

OpenFace та SSD	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Ні	0,43	0,1	2,87
euclidean	Ні	0,93	0,55	2,87
euclidean_l2	Ні	0,93	0,55	2,87

За результатами таблиць 3.18 – 3.21 можна сказати, що найрезультативніша зв’язка це ArcFace та OpenCV. ArcFace з OpenCV показує точний результат за менший час завдяки своїй архітектурі. Поганим результатом вийшли зв’язки з детектором SSD. Всі результати були вище порогового значення.

Таблиця 3.18 – Результати ArcFace та OpenCV

ArcFace та OpenCV	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Так	0,49	0,68	3,86
euclidean	Так	3,88	4,15	3,67
euclidean_l2	Так	0,99	1,13	3,68

Таблиця 3.19 – Результати ArcFace та RetinaFace

ArcFace та RetinaFace	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Так	0,49	0,68	6,65
euclidean	Так	3,8	4,15	6,76
euclidean_l2	Так	0,99	1,13	6,62

Таблиця 3.20 – Результати ArcFace та MTCNN

ArcFace та MTCNN	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Так	0,52	0,68	4,31
euclidean	Так	3,40	4,15	4,2
euclidean_l2	Так	1,02	1,13	4,31

Таблиця 3.21 – Результати ArcFace та SSD

ArcFace та SSD	**Результат порівняння (Так/Ні)**	**Відстань**	**Поріг**	**Час, с**
cosine	Ні	1,00	0,68	3,44
euclidean	Ні	11,62	4,15	3,43
euclidean_l2	Ні	1,41	1,13	3,39

У висновку, за результатами всіх тестів можна зазначити, що детектор MTCNN показав найкращий результат серед усіх. Він легко комбінується з іншими моделями нейронних мереж. Детектор OpenCV також добре поєднується з мережами з меншою кількістю параметрів як ArcFace та FaceNet. Детектор SSD показав найгірші результати. Детектор потрібно навчати та покращувати.

3.4 Перспективи подальшої роботи

Розроблений застосунок відповідає вимогам поставленої задачі і вміло визначає ступені схожості облич на еталонних зображеннях.

Загальна оцінка роботи застосунку свідчить про те, що він показав добрі результати та відповідає цілям розробки. Навіть при виявленні деяких недоліків під час тестування, потенціал для подальшого вдосконалення додатку виявився більшим за його обмеження.

Серед можливих шляхів поліпшення системи важливо відзначити необхідність процесу навчання нейронної мережі та подальшого покращення детекторів розпізнавання облич.

Незважаючи на існуючі обмеження та недоліки, необхідно підкреслити, що головна мета розробки була досягнута, і система працює відповідно до

початкових планів та очікувань. Це свідчить про важливий крок у розвитку системи розпізнавання та визначення ступеню схожості облич.

Застосунок відкриває широкий спектр можливостей для подальшої роботи у галузі розпізнавання облич за допомогою нейронних мереж. Досягнення в технологіях та методах в цій області можуть призвести до створення більш точних, гнучких та багатофункціональних систем для визначення ступеню схожості облич.

ВИСНОВКИ

У рамках кваліфікаційної роботи була розроблена система розпізнавання та визначення ступеню схожості облич (додаток А).

Виконано всі поставлені задачі, а саме:

– проаналізовані методи визначення ступеню схожості облич, а саме нейронні мережі VGG-Face, DeepFace, FaceNet, OpenFace, ArcFace та детектори облич SSD, MTCNN, RetinaFace та OpenCV, що дозволило в парах нейронна мережа та детектор порівняти між собою якість та швидкість розпізнавання;

– розроблено методику виявлення та верифікації обличчя за допомогою згорткових нейронних мереж та детекторів;

– визначено необхідні інструментальні засоби для створення застосунку;

– виконано всі етапи розроблення застосунку для визначення ступеню схожості облич, на кожному етапі здійснено різноманітні програмні та алгоритмічні оптимізації, реалізовано гнучкий метод роботи застосунку, його взаємодії з користувачем;

– проведено тестування розробленого застосунку, за результатами якого було визначено особливості роботи реалізованого методу, умови використання та його ефективність, яка виявилася на даному етапі задовільною, але водночас надавала розуміння необхідности ширших випробувань та можливого подальшого доопрацювання;

– визначено перспективи подальшої роботи, які виходять з отриманих результатів тестування та теоретичних знань в досліджуваній області.

За результатами всіх тестів можна зазначити, що детектор MTCNN показав найкращий результат серед усіх. У парі з мережею VGG-Face та рівнянням euclidean MTCNN показав коефіцієнт відстані 0,46 та час 4,13 секунди. З мережею DeepFace детектор MTCNN показав коефіцієнт відстані 0,14 та час 27,25 секунди. Мережа OpenFace та рівняння euclidean показало

коефіцієнт відстані 0,51 та час у 3,67 секунди. Він легко комбінується з іншими моделями нейронних мереж та показує найкращу ступінь розпізнавання та час.

Детектор RetinaFace за якістю розпізнавання відстає на декілька сотих за MTCNN. З мережею DeepFace та рівнянням cosine, RetinaFace має коефіцієнт відстані 0,16 проти 0,14 у MTCNN та час 29,85 секунди проти 27,25 секунди у MTCNN аналогічно. RetinaFace показала найкращу пару з мережею VGG-Face серед інших детекторів, коефіцієнт відстані 0,24 та час 6,9 секунди. Недоліками детектора є великий час розпізнавання в усіх тестах з усіма мережами відносно інших пар.

Детектор OpenCV також добре поєднується з мережами з меншою кількістю параметрів як ArcFace та FaceNet. При мережі FaceNet та рівнянні cosine OpenCV показує коефіцієнт відстані 0,24 та час у 4,86 секунди. З мережею ArcFace та рівнянням cosine OpenCV показав коефіцієнт відстані 0,49 та час у 3,86 секунди.

Детектор SSD показав найгірші результати. Значення відстані завжди була більше зазначеного порогу. У мережі VGG-Face та рівнянні cosine детектор SSD мав коефіцієнт відстані 0,46 при коефіцієнту порогу 0,4. SSD менше точний та гнучкий до розпізнавання різних кутів та виразів обличчя як наприклад RetinaFace.

Результати дослідження апробовано у вигляді тез доповіді під час XXXVI Міжнародної науково-практичної конференції «Modern problems and the latest theories of development» [45].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Face Recognition Technology (FERET). URL: https://datasciencetoday.net/index.php/en-us/datasets/[33-face-recognition-technology-feret (дата звернення 25.09.2023).

2. Dev, S., & Patnaik, T. (2020, September). Student attendance system using face recognition. In *2020 international conference on smart electronics and communication (ICOSEC)* (pp. 90-96). IEEE.

3. Ayyappan, S., & Matilda, S. (2020, July). Criminals and missing children identification using face recognition and web scrapping. In *2020 International Conference on System, Computation, Automation and Networking (ICSCAN)* (pp. 1-5). IEEE.

4. William, I., Rachmawanto, E. H., Santoso, H. A., & Sari, C. A. (2019, October). Face recognition using facenet (survey, performance test, and comparison). In *2019 fourth international conference on informatics and computing (ICIC)* (pp. 1-6). IEEE.

5. I. Masi, Y. Wu, T. Hassner and P. Natarajan, "Deep Face Recognition: A Survey," *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, Parana, Brazil, 2018, pp. 471-478.

6. Deng, J., Guo, J., Xue, N., & Zafeiriou, S. (2019). Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 4690-4699).

7. Bonam, J., Burra, L. R., Godavarthi, R. S. V., Jagabattula, D., Eda, S., & Gogulamudi, S. (2022). Multiple Face Recognition System Using OpenFace. In *Intelligent Computing and Applications: Proceedings of ICDIC 2020* (pp. 339-349). Singapore: Springer Nature Singapore.

8. How to implement Face Recognition using VGG Face in Python 3.7 and Tensorflow 2.0. URL: https://medium.com/analytics-vidhya/how-to-implement-

face-recognition-using-vgg-face-in-python-3-8-and-tensorflow-2-0-a0593f8c95c3 (дата звернення 26.09.2023).

9. New Facial Recognition Technology Trends To Boom In The Future. URL: https://www.lystloc.com/blog/7-new-facial-recognition-technology-trends-to-boom-in-the-future/ (дата звернення 27.09.2023).

10. Chicco, D. (2021). Siamese neural networks: An overview. *Artificial neural networks*, 73-94.

11. Lei, J., Gu, Y., Xie, W., Li, Y., & Du, Q. (2022). Boundary extraction constrained siamese network for remote sensing image change detection. *IEEE Transactions on Geoscience and Remote Sensing*, *60*, 1-13.

12. F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in Proceedings of the IEEE conference on computer vision and pattern recognition. IEEE, 2015, pp. 815-823.

13. Turk, M. A., & Pentland, A. P. (1991, January). Face recognition using eigenfaces. In *Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition* (pp. 586-587). IEEE Computer Society.

14. Kiran, T. A., Reddy, N. D. K., Ninan, A. I., Krishnan, P., Aravindhar, D. J., & Geetha, A. (2020, September). Pca based facial recognition for attendance system. In *2020 International Conference on Smart Electronics and Communication (ICOSEC)* (pp. 248-252). IEEE.

15. Facial recognition market size worldwide from 2019 to 2032. URL: https://www.statista.com/statistics/1153970/worldwide-facial-recognition-revenue/ (дата звернення 28.09.2023).

16. J. G. Cavazos, P. J. Phillips, C. D. Castillo and A. J. O’Toole, "Accuracy Comparison Across Face Recognition Algorithms: Where Are We on Measuring Race Bias?," in *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 3, no. 1, pp. 101-111, Jan. 2021.

17. Another Arrest, and Jail Time, Due to a Bad Facial Recognition Match. URL: https://www.nytimes.com/2020/12/29/technology/facial-recognition-misidentify-jail.html (дата звернення 28.09.2023).

18. Anwarul, S., & Dahiya, S. (2020). A comprehensive review on face recognition methods and factors affecting facial recognition accuracy. *Proceedings of ICRIC 2019: Recent Innovations in Computing*, 495-514.

19. Massoli, F. V., Amato, G., & Falchi, F. (2020). Cross-resolution learning for face recognition. *Image and Vision Computing*, 99, 103927.

20. Shah, A., Bangash, J. I., Khan, A. W., Ahmed, I., Khan, A., Khan, A., & Khan, A. (2022). Comparative analysis of median filter and its variants for removal of impulse noise from gray scale images. *Journal of King Saud University-Computer and Information Sciences*, *34*(3), 505-519.

21. Chen, Y., Wang, Z., & Zhang, K. (2013). Approximations for modulus of gradients and their applications to neighborhood filters. *Frontiers of Mathematics in China*, 8, 761-782.

22. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., Gadetska S., and Al-Dhaifallah M. (2023) Statistical data analysis models for determining the relevance of structural image descriptions, *IEEE Access*, 11, pp. 126938-126949.

23. Гороховатський В., Творошенко І., Сидоренко Д. (2021) Класифікація зображень із використанням кластерного подання, *Міжн. наук. симпозіум Інтелектуальні рішення-С. Обчислювальний інтелект. Теорія прийняття рішень: праці міжн. наук. симп. (Вересень 29, 2021)*. Київ-Ужгород, С. 44-45.

24. 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.

25. 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.

26. 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.

27. Gorokhovatskyi, V., Peredrii, O., Tvoroshenko, I., & Markov, T. (2023). Матриця відстаней для множини компонентів структурного опису як інструмент для створення класифікатора зображень, *Advanced Information Systems*, 7(1), С. 5–13.

28. Pomazan, V., Tvoroshenko, I., & Gorokhovatskyi, V. (2023). Development of an application for recognizing emotions using convolutional neural networks, *International Journal of Academic Information Systems Research*, 7(7), pp. 25-36.

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

30. Liu, Y., & Chen, J. (2021). Unsupervised face frontalization for pose-invariant face recognition. *Image and Vision Computing*, 106, 104093.

31. Elharrouss, O., Almaadeed, N., Al-Maadeed, S., & Khelifi, F. (2022). Pose-invariant face recognition with multitask cascade networks. *Neural Computing and Applications*, 1-14.

32. Wang, M., & Deng, W. (2020). Deep face recognition with clustering based domain adaptation. *Neurocomputing*, *393*, 1-14.

33. Shakeel, M. S., & Lam, K. M. (2019). Deep-feature encoding-based discriminative model for age-invariant face recognition. *Pattern Recognition*, *93*, 442-457.

34. Anwarul, S., & Dahiya, S. (2020). A comprehensive review on face recognition methods and factors affecting facial recognition accuracy. *Proceedings of ICRIC 2019: Recent Innovations in Computing*, 495-514.

35. Kortli, Y., Jridi, M., Al Falou, A., & Atri, M. (2020). Face recognition systems: A survey. *Sensors*, *20*(2), 342.

36. Sharma, S., Bhatt, M., & Sharma, P. (2020, June). Face recognition system using machine learning algorithm. In *2020 5th International Conference on Communication and Electronics Systems (ICCES)* (pp. 1162-1168). IEEE.

37. Bounds, A. D., & Girkin, J. M. (2021). Early stage dental caries detection using near infrared spatial frequency domain imaging. *Scientific Reports*, 11(1), 2433.

38. Kola, D. G. R., & Samayamantula, S. K. (2021). A novel approach for facial expression recognition using local binary pattern with adaptive window. *Multimedia Tools and Applications*, *80*, 2243-2262.

39. Pomazan V., Tvoroshenko I., and Gorokhovatskyi V. (2023) Handwritten character recognition models based on convolutional neural networks, *International Journal of Academic Engineering Research*, 7(9), pp. 64-72.

40. Tvoroshenko I., Gorokhovatskyi V., Kobylin O., and Tvoroshenko A. (2023) Application of deep learning methods for recognizing and classifying culinary dishes in images, *International Journal of Academic and Applied Research*, 7(9), pp. 57-70.

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

42. Gorokhovatskyi V., Tvoroshenko I. (2023) Identification of visual objects by the search request. *International scientific symposium «INTELLIGENT SOLUTIONS-S». Computational intelligence (results, problems and perspectives). Decision making theory: proceedings of the international symposium*, September 28, 2023, Kyiv-Uzhorod, Ukraine, pp. 25-27.

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*, 6(3), С. 5–12.

44. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., Gadetska S., and AI-Dhaifallah M. (2023) Statistical data analysis models for determining the relevance of structural image descriptions, IEEE Access, 11, pp. 126938-126949.

45. Тарасов Д. (2023) Аналіз сучасних тенденції у сфері технологій розпізнавання облич, *Abstracts of XXXVI International Scientific and Practical Conference «Modern problems and the latest theories of development», (September 11 – 13, 2023). Munich, Germany*, pp. 269-272.