Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Система розпізнавання біометричних даних обличчя
з використанням технологій машинного навчання

(тема)

Виконав:
здобувач 2025 року навчання,
групи СКСм-23-1
Мілька Я.Ю.
(прізвище, ініціали)

Спеціальність
123 – Комп'ютерна інженерія
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Спеціалізовані
комп'ютерні системи
(повна назва освітньої програми)

Керівник: доц. Ларченко Л.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Чумаченко С.В.
(підпис) (прізвище, ініціали)

2025 р

Харківський національний університет радіоелектроніки

Факультет Комп’ютерної інженерії та управління

Кафедра Автоматизації проектування обчислювальної техніки

Рівень вищої освіти другий (магістерський)

Спеціальність 123 – Комп’ютерна інженерія
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Спеціалізовані комп’ютерні системи
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

“____” _____ 20__ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Мільці Ярославу Юрійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Система розпізнавання біометричних даних обличчя з використанням технологій машинного навчання

затверджена наказом по університету від "__08_" __11__2024 р. № _1189 Ст_____

2. Термін подання студентом роботи до екзаменаційної комісії 24.01.2025

3. Вихідні дані до роботи

Операційна система розробки – MacOS

Середовище розробки – PyCharm Professional

Розробка модулів для біометричної ідентифікації користувачів

Мова програмування - Python

4. Перелік питань, що потрібно опрацювати в роботі

1 Огляд методів та систем біометричного розпізнавання облич

2 Алгоритмічні рішення біометричного розпізнавання обличчя

3 Програмна реалізація системи біометричного розпізнавання облич

4. Тестування розробленої системи

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) _____
18 слайдів

6. Консультанти розділів роботи (заповнюється 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

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	02.09.2024 - 02.09.2024	
2	Аналіз поставленої задачі	03.09.2024 - 10.09.2024	
3	Дослідження існуючих методів вирішення поставленої задачі	11.09.2024 - 18.09.2024	
4	Дослідження методів заснованих на нейронних мережах	19.09.2024 - 10.10.2024	
5	Виявлення проблем і недоліків	11.11.2024 – 25.11.2024	
6	Дослідження можливих рішень виявлених проблем	26.11.2024 - 03.12.2024	
7	Оформлення пояснювальної записки	04.12.2024 - 18.12.2024	
8	Оформлення графічного матеріалу	19.12.2024 - 25.12.2024	
9	Перевірка виконаного проекту керівником	26.12.2024 -05.01.2025	
10	Захист роботи	24.01.2025	

Дата видачі завдання 02 вересня 2024 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Ларченко Л.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка містить 74 сторінок, 25 рисунків, 16 джерел за переліком посилань.

РОЗПІЗНАВАННЯ ОБЛИЧ, МАШИННЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ, CNN, ОПТИМІЗАЦІЯ, ПРИВАТНІСТЬ

В кваліфікаційній роботі проаналізовано методи та технології машинного навчання для систем розпізнавання біометричних даних обличчя. Значна увага приділена застосуванню згорткових нейронних мереж, що дозволяє підвищити точність та ефективність ідентифікації особи.

Об’єктом дослідження є системи біометричного розпізнавання обличчя, предметом дослідження - моделі та методи ідентифікації обличчя користувача за біометричними характеристиками. Досліджено загальні принципи побудови нейронних мереж для розпізнавання облич з врахуванням специфіки завдання і використанням відповідних методів і архітектури.

В роботі розроблено систему розпізнавання біометричних даних обличчя з використанням технологій машинного навчання. Програмна реалізація системи розпізнавання розрблена за допомогою мови програмування Python з використанням фреймворку Streamlit, що являє собою веб-застосунок з платформою машинного навчання TensorFlow, середовищем розробки є PyCharm.

Результати роботи можуть бути використані для подальшого вдосконалення біометричних систем в комерційних і державних організаціях.

ABSTRACT

The explanatory note contains 74 pages, 25 figures, 16 sources by the list of references.

FACIAL RECOGNITION, MACHINE LEARNING, NEURAL NETWORKS, CNN, OPTIMIZATION, PRIVACY

The qualification work analyzes the methods and technologies of machine learning for biometric facial data recognition systems. Considerable attention is paid to the use of convolutional neural networks, which allows to increase the accuracy and efficiency of personal identification.

The object of the study is biometric facial recognition systems, the subject of the study is models and methods of identifying a user's face by biometric characteristics.

A number of optimization techniques and methods for increasing data privacy in biometric systems are considered. As a result, general principles for building neural networks for facial recognition are obtained, taking into account the specifics of the task and using appropriate methods and architecture.

The work develops a biometric facial data recognition system using machine learning technologies.

The results of the work can be used to further improve biometric systems in commercial and government organizations.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ANN (Artificial Neural Network) – штучна нейронна мережа

CNN (Convolutional Neural Network) – згорткова нейронна мережа

LBP (Local Binary Pattern) – метод локальних бінарних шаблонів

PCA (Principal Component Analysis) – метод головних компонент

SVM (Support Vector Machine) – метод опорних векторів

DCT (Discrete Cosine Transform) – дискретне косинусне перетворення

HOG (Histogram of Oriented Gradients) – гістограма орієнтованих градієнтів

SIFT (Scale-Invariant Feature Transform) – перетворення, інваріантне до масштабу.

ВСТУП

Актуальність дослідження систем розпізнавання біометричних даних обличчя з використанням технологій машинного навчання обумовлена зростаючими потребами в підвищенні рівня безпеки, а також ефективності ідентифікації особистості в різних сферах діяльності. У сучасних умовах цифровізації та глобальної взаємодії, проблема захисту персональних даних і надійності ідентифікаційних процесів стає критично важливою. Традиційні методи аутентифікації, такі як паролі та картки доступу, виявляються вразливими до кіберзагроз, фальсифікацій та шахрайства, що викликає необхідність у впровадженні більш складних і надійних технологій. Системи розпізнавання обличчя, будучи однією з найбільш перспективних біометричних технологій, пропонують рішення для багатьох задач: від забезпечення безпеки на державному рівні до застосування в повсякденному житті, як, наприклад, мобільні пристрої та системи «розумного дому».

Метою кваліфікаційної роботи є розробка системи розпізнавання біометричних даних обличчя з використанням технологій машинного навчання. Важливо підкреслити, що сучасні досягнення в галузі машинного навчання, зокрема глибоких нейронних мереж, дозволяють значно підвищити точність і швидкість розпізнавання, забезпечуючи адаптивність систем до змін зовнішніх умов, таких як освітлення або кут зйомки.

Тема роботи є не лише актуальною, але й необхідною для подальшого розвитку інтелектуальних систем, які можуть інтегруватися в різні аспекти діяльності суспільства: від правоохоронних органів до комерційних організацій, що забезпечує її високу наукову та практичну значущість.

1 ОГЛЯД МЕТОДІВ ТА СИСТЕМ БІОМЕТРИЧНОГО РОЗПІЗНАВАННЯ ОБЛИЧ

В даному розділі розглянуто методи біометричного розпізнавання обличчя, зокрема, метод на основі геометричних характеристик обличчя, метод з використанням глибокого навчання на основі нейронної мережі та метод вектора головних компонентів, розглянуто переваги та недоліки зазначених методів, а також проведено огляд існуючих систем розпізнавання обличчя, сформульовано мету та постановку завдання.

1.1. Основи біометричного розпізнавання обличчя

Обличчя людини виділяється як візуальна риса, яка вирізняє кожну людину. У нашому повсякденному житті обличчя є, мабуть, найвідомішою та загальновизнаною біометричною характеристикою. З появою фотографії, як державні організації, так і приватні установи почали накопичувати колекції фотографій облич, взятих із різних документів, що посвідчують особу, таких як посвідчення особи, паспорти. Дані компіляції виявилися корисними при проведенні розслідувань, слугуючи довідковими базами даних для зіставлення та порівняння зображень облич особистостей, будь то підозрювані, свідки чи жертви. Крім того, розвиток цифрових фотоапаратів і смартфонів надалі зробило зйомку зображень обличчя поширеним завданням, а отже, тепер зображення можна швидко поширювати завдяки постійному розширенню охоплення соціальних мереж [1].

Протягом кількох десятиліть прогрес у галузях електроніки та інформатики революціонізував та забезпечив доступ до передових технологій для значної частини населення. Цей прогрес зробив технологічні пристрої високого рівня доступнішими, ніж будь-коли раніше. Однією зі

сфер, де досягнення особливо помітні, є сфера біометричних методів, які поступово замінили традиційні рішення, засновані на знаннях, таких як паролі або PIN-коди, а також стратегії, засновані на володінні, як-от ідентифікаційні картки або значки [2]. У різних програмах, таких як онлайн-платежі та електронна комерція, автентифікація смартфонів, біометричні паспорти та прикордонний контроль, широко використовуються різні біометричні системи. Серед численних доступних біометричних характеристик людське обличчя, незважаючи на те, що воно не є бездоганним ідентифікатором порівняно зі скобами, такими як райдужка ока чи відбитки пальців, має явні переваги, які позиціонують його як один із найпопулярніших і перспективних методів розпізнавання особистості.

Розпізнавання обличчя відноситься до передової технології, яка призначена для розпізнавання та автентифікації особи за допомогою різних форм візуального медіа, наприклад зображень, відео або будь-якого візуального представлення її обличчя. Цей метод ідентифікації, зазвичай, використовується для отримання доступу до програм, систем або служб, функціонуючи подібно до сканера обличчя. Сканер підпадає під категорію біометричної ідентифікації, покладаючись на унікальні характеристики обличчя для ефективної перевірки особистості на основі чітких біометричних шаблонів обличчя та даних.

Технологія збирає повний набір ексклюзивної біометричної інформації, що включає риси обличчя та вирази, яка потім використовується для ідентифікації, верифікації та автентифікації особи. Завдяки цьому повний набір біометрії забезпечує підвищений рівень безпеки та точності процесів ідентифікації особистості. Процес ідентифікації облич потребує пристроїв, обладнаних цифровими фотографічними технологіями, що дозволяють захоплювати та збирати необхідні зображення та дані для створення та зберігання унікального біометричного портрета обличчя людини. На відміну від інших методів та засобів ідентифікації, які включають паролі, підтвердження електронної пошти або розпізнавання відбитків пальців,

використання біометричного розпізнавання обличчя ґрунтується на чітких математичних і динамічних моделях, які функціонують подібно до сканера обличчя. Такий підхід забезпечує високий рівень безпеки та ефективності.

Основна мета розпізнавання обличчя полягає у проведенні аналізу вхідного зображення та зіставлення його з відповідними даними з колекції навчальних зображень, що зберігаються в базі даних.

Однак, одна з головних проблем полягає у виконанні цього процесу в режимі реального часу, що залишається проблемою для постачальників біометричного програмного забезпечення для розпізнавання обличчя. Технологія розпізнавання обличчя використовується в різних сферах для виконання різних функцій в залежності від її використання.

Системи розпізнавання обличчя працюють шляхом захоплення зображень із пристроїв-камер у двовимірному або тривимірному вигляді залежно від конкретних характеристик пристрою. Далі системи порівнюють відповідну інформацію, що отримано із зробленого зображення в режимі реального часу з базою даних фотографій або відео, забезпечуючи вищий рівень надійності та безпеки порівняно зі статичними зображеннями. Варто зазначити, що цей процес біометричного розпізнавання обличчя вимагає підключення до Інтернету, оскільки база даних зберігається не на самому пристрої захоплення, а на віддалених серверах.

Під час порівняння облич система розпізнавання використовує математичний аналіз з метою ретельної перевірки вхідного зображення, не залишаючи місця для помилок. За допомогою цього аналізу система перевіряє, чи відповідають біометричні дані, отримані з вхідного зображення, призначеному користувачеві, який запитує доступ до системи.

Перший сценарій передбачає початкову реєстрацію користувача в системі розпізнавання обличчя, коли система фіксує обличчя людини та пов’язує його з певною особою, таким чином зберігаючи його в системі. Цей конкретний процес зазвичай називають «цифровою адаптацією» з розпізнаванням обличчя. Другий варіант виникає, коли користувач

проходить аутентифікацію перед реєстрацією. Під час цього процесу вхідні дані, зняті камерою, порівнюються з наявними даними, що зберігаються в базі даних. Якщо знайдено збіг між захопленим обличчям при ідентифікації, і вже зареєстрованою особою, користувачеві надається доступ до системи за допомогою облікових даних. На рис. 1.1 представлена концепція біометричного розпізнавання обличчя [3].

Рисунок 1.1 – Концепція біометричної ідентифікації людини

1.2. Огляд методів виявлення та обробки зображення для біометричного розпізнавання обличчя

На сьогоднішній день існує кілька методик, спрямованих на розпізнавання облич. Традиційний підхід передбачає визначення чітких рис обличчя, а саме, ключових моментів, таких як відстань між очима, ширина щелепи та носа та інших частин обличя. Аналізуючи розмір, форму та відносне розташування цих рис, створюється відбиток обличчя, який зберігається в базі даних у числовому коді. Далі здійснюється порівняння числового коду з еталонними прикладами з метою ідентифікації особи людини на зображенні. В даний час традиційний метод отримав широке

застосування завдяки своїй універсальності. Він не потребує спеціального обладнання, може працювати на відстані, використовуючи алгоритми для покращення зображень обличчя з низькою роздільною здатністю, і вимагає відносно простого отримання фотографій для розпізнавання. Тим не менш, цей підхід має кілька недоліків, включаючи сприйнятливість до оклюзії, повороту та зміни виразу обличчя.

Важливо зазначити, що спочатку науковці зосереджувалися на розпізнаванні облич за контрольованих обставин, де традиційні підходи давали задовільні результати. Однак напрямок досліджень змістився у бік розпізнавання облич у вільному середовищі. Отже, останніми роками методи машинного навчання стали найкращим вибором майже в усіх програмах розпізнавання облич, оскільки вони пропонують більш широкі можливості розробки та дають результати вищої якості. Запровадження глибокого навчання на початку 2010-х років революціонізувало сферу комп'ютерного зору, значно підвищивши продуктивність у багатьох сферах, таких як класифікація зображень і розпізнавання об'єктів. Глибоке навчання зіграло вирішальну роль у досягненні вражаючих показників точності навіть за наявності складних факторів, які перешкоджають функціональності програми [4]. Як наслідок, з'явилася велика кількість згорткових нейронних мереж, включаючи DeepFace, DeepIDs, FaceNet та SphereFace. Дослідження показують, що ці мережі демонструють схожий рівень точності розпізнавання.

1.2.1 Метод, заснований на геометричних характеристиках обличчя

Більшість технік, які використовувалися раніше, в основному, базувалися на геометричних принципах, зосереджуючись на атрибутах обличчя людей. Ці методи використовували різні вимірювання між ключовими точками обличчя та варіаціями кутів, створених векторами відстані. Наприклад, зазвичай, використовувалися такі фактори, як розмір голови, міжочна відстань і чіткість контуру підборіддя.

Прогрес у цьому підході призвів до використання методу на основі архітектури динамічного зв’язку. Архітектура динамічних посилань, на основі якої створено систему відповідності графів, є помітним удосконаленням [5]. Одним з конкретних прикладів дуже успішної системи, що використовує цей підхід, є система Elastic Bunch Graph Matching.

Типове представлення місцевих рис обличчя включає вейвлет-коефіцієнти для різних масштабів і поворотів на основі попередньо визначеного вейвлета. Ці локально оцінені коефіцієнти є стійкими до змін освітленості, спотворень, поворотів і масштабування зображень обличчя.

На відміну від традиційного методу, в якому здійснюється призначення окремого коефіцієнта кожному вузлу, Elastic Bunch Graph Matching включає представлення зв’язку графа, що складається з набору вейвлет-коефіцієнтів, кожен з яких отримано з іншого зображення обличчя [6]. Ця складна техніка дозволяє більш повно аналізувати риси обличчя. Для того, щоб точно визначити положення обличчя, EBGM використовує попередні знання, отримані в результаті обробки змін положення. Вивчаючи перетворення вейвлет-коефіцієнтів під час зміни положення обличчя, EBGM ефективно фіксує тонкі зміни в орієнтації обличчя. Це, в свою чергу, сприяє більш точному оцінюванню пози обличчя. Широке застосування підходу EBGM свідчить про його універсальність та ефективність, який успішно використовується в різних сферах, таких як виявлення та видалення облич, оцінка пози, гендерна класифікація, розпізнавання на основі ескізу та загальне розпізнавання об’єктів. Це підкреслює адаптивність системи EBGM та її здатність вирішувати різноманітні проблеми комп’ютерного зору [7, 8].

Подальший прогрес у розпізнаванні обличчя передбачав використання прихованих моделей Маркова. Замість того, щоб точно визначити риси обличчя, цей підхід використовує піксельні смуги, які охоплюють лоб, очі, ніс, рот і підборіддя. Приховані моделі Маркова (ПММ) набули широкого поширення в різних сферах завдяки своїй ефективності в моделюванні різноманітних процесів і полегшенні розпізнавання образів. ПММ чудово

вловлюють просторово-часові характеристики сигналів, завдяки чому вони широко використовуються в таких сферах, як розпізнавання мовлення та, останнім часом, аналіз зображень обличчя. ПММ складається з набору N станів, які переходять з одного в інший. У будь-якому заданому сценарії система перебуває в точно визначеному стані, а в звичайних моделях Маркова першого порядку цей стан залежить виключно від поточного стану. Крім того, при переході з одного стану в інший генерується спостережуваний символ, що представляє фізичний сигнал, отриманий з виходу змодельованої системи. Важливо підкреслити, що вихід моделі може бути дискретним або безперервним, залежно від характеру програми. Крім того, існують ПММ, які використовують однаковий набір символів для всіх станів, забезпечуючи таким чином узгодженість у всій моделі.

На рис. 1.2 представлений приклад використання моделі ПММ для використання у технологіях біометричного розпізнавання обличчя.

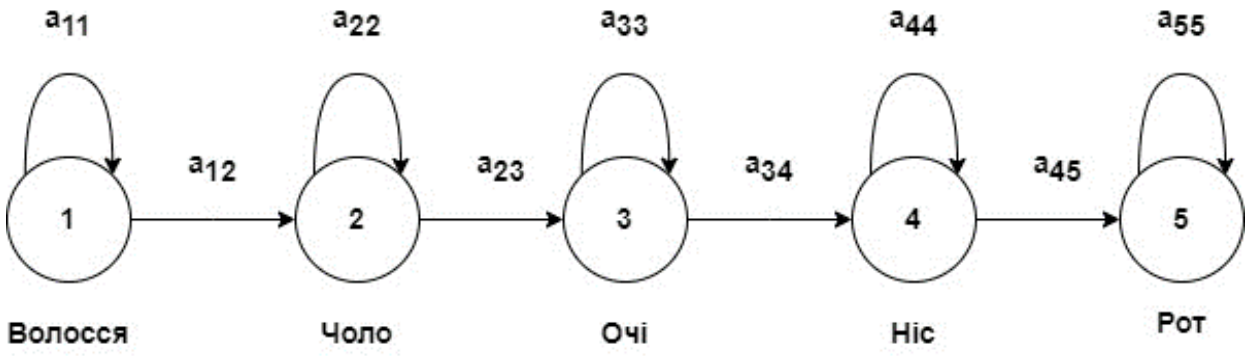

Рисунок 1.2 – Використання ПММ для технології розпізнавання обличчя

Двовимірні марковські моделі забезпечують чіткий підхід до зображення, спотворення зображення та просторового співвідношення між різними регіонами. На відміну від одновимірних лінійних моделей ПММ, які зосереджуються окремо на горизонтальних або вертикальних аспектах, двовимірні моделі Маркова враховують обидва напрямки одночасно. Цей

комплексний підхід дозволяє більш глибоко зрозуміти спотворення зображення та відносне розташування різних областей на зображенні. Для спрощення обчислювальної складності використовуються псевдодвовимірні ПММ. Дана модель містить кілька лінійних вертикальних моделей на нижньому рівні, а також одну лінійну горизонтальну модель на верхньому рівні. Виходи моделей нижнього рівня служать вхідними даними для моделі верхнього рівня. Кожен стан моделі верхнього рівня складається з послідовності станів, отриманих з відповідної моделі нижнього рівня. Слід зазначити, що моделі нижчого рівня працюють незалежно одна від одної. Спочатку моделі верхнього рівня були переважно вертикальними, але подальші дослідження розширили моделі, включивши горизонтальні компоненти. Це вдосконалення дозволило вертикальним моделям нижчого рівня враховувати варіації рівня очей. У результаті псевдодвовимірна модель ефективно справляється з локальними деформаціями та точно відображає взаємне розташування різних областей зображення. Використання 2D-моделі Маркова, зокрема її псевдо-2D-ітерації, дає сприятливі результати для розуміння спотворення зображення та взаємозв’язків між різними розділами. Ця модель гарантує, що область, яка охоплює певну функцію, наприклад око, ніколи не перекривається з областю, зайнятою іншою особливістю, такою як рот. Таким чином, досягається більш глибоке розуміння локальних деформацій, що полегшує точне відображення відповідних областей на аналізованому зображенні. В області моделей зіставлення шаблонів початкова ініціалізація моделі відіграє критичну роль. Моделі ініціалізуються з використанням усіх зображень з навчального набору. Кожен клас пов’язаний із певною моделлю та зображенням, яке її представляє.

Однак модель зіставлення шаблонів має помітне обмеження, яке полягає в тому, що їй бракує дискримінаційної здатності. Алгоритм навчання в першу чергу зосереджується на максимізації реакції кожної моделі на відповідний клас, без ефективної мінімізації відповідей на інші класи.

1.2.2 Метод з використанням глибокого навчання

Структура багатошарової нейронної мережі складається з взаємопов’язаних шарів. Кожен шар складається з нейронів, які отримують вхідні дані від нейронів попереднього шару та передають вихідні дані нейронам наступного шару. Маючи кілька рівнів прийняття рішень, НМ здатна точно апроксимувати багатовимірні функції. Однак, якщо НМ має лише один рівень прийняття рішень, то він може створювати лише лінійні розділові поверхні, що значно обмежує можливості рівня прийняття рішень щодо вирішення проблем. Наприклад, однорівнева мережа прийняття рішень є неефективною у вирішенні проблем, «що виключає або». Навпаки, НМ з нелінійною функцією активації та двома рівнями прийняття рішень може генерувати опуклі області в просторі рішень. Крім того, за допомогою трьох рівнів рішень НМ може охоплювати області різної складності, включаючи невипуклі. Процес навчання нейронної мережі передбачає використання алгоритму зворотного поширення. Цей алгоритм фокусується на коригуванні вагових коефіцієнтів за допомогою методу градієнтного спуску. Основною метою є мінімізація загальної похибки мережі. Поширюючи помилки, представлені у вигляді значень корекції ваги, у зворотному напрямку від входів до виходів уздовж з’єднань між нейронами, алгоритм забезпечує ефективне навчання. Архітектура та функціональність багатошарової нейронної мережі залежать від різних факторів. Ці фактори включають кількість рівнів прийняття рішень, конкретну проблему та тип використовуваної функції активації. Одним з корисних застосувань одношарової нейронної мережі є автоасоціативна пам’ять. Ця техніка дозволяє навчити мережу реконструювати представлені зображення. Інший підхід до використання нейронних мереж для обробки зображень полягає в прямій класифікації зображень. У цій методології вхідні дані можуть складатися або з самого зображення, або з набору ключових характеристик, вилучених із зображення. Вихідний нейрон із найвищим рівнем активності вказує на клас, до якого належить розпізнане зображення. У контексті

розпізнавання зображень, якщо нейрон із найвищим рівнем активності, як правило, перший нейрон у мережі, означає, що вхідне зображення належить до розпізнаного класу. Однак, якщо рівень активності зменшується нижче певного порогу, зображення вважається таким, що не належить до жодного відомого класу. Цей процес називається «навчання з учителем», який полягає в зіставленні зображень із відповідними класами. Даний підхід зазвичай використовується під час виконання завдань контролю доступу, які включають обмежену кількість облич, особливо в сценаріях, де метою є розпізнавання облич людей із зображень. Однак із збільшенням кількості окремих класів експоненціально зростає як час, необхідний для навчання, так і продуктивність мережі.

У сфері класифікації зображень особливим підходом, який має велике значення, є використання багатошарових нейронних мереж (БНМ), які враховують різні риси обличчя, такі як просторові відстані між різними компонентами, такими як ніс, рот і очі. Крім того, існують гібридні системи, які поєднують БНМ з іншими моделями, такими як модель Маркова. У класичній реалізації БНМ нейронні зв’язки встановлюються між шарами з метою представлення зображення як одновимірного вектору, незважаючи на притаманну двовимірність самого зображення. Завдяки цьому процесу ключові атрибути обличчя фіксуються та аналізуються, що дозволяє виконувати точну класифікацію та розпізнавання. Цей підхід допомагає подолати обмеження, пов’язані з порівнянням зображень, створюючи більш ефективні та масштабовані системи розпізнавання облич.

На рис. 1.3 представлений приклад архітектури згорткової нейронної мережі, яка використовується в сфері розпізнавання об’єктів [9].

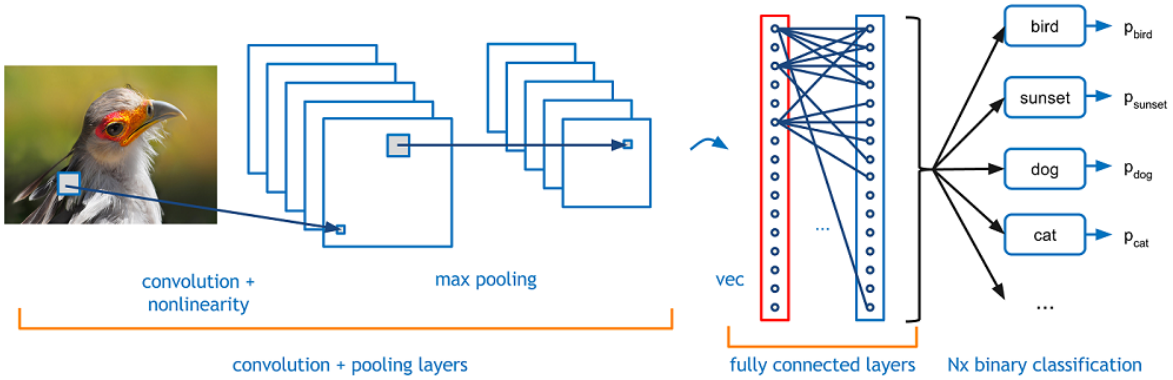

Рисунок 1.3 – Типова архітектура згорткової нейронної мережі

Розглядаючи концепцію архітектури згорткової нейронної мережі (ЗНМ), важливо зазначити, що дана мережа складається з кількох шарів із численними взаємопов'язаними площинами. Нейронні зв'язки між наступними шарами спеціально розроблені для імітації структури зорової кори головного мозку людини. Кожен шар у ЗНМ має локалізовану модель з'єднання, тобто нейрони одного шару з'єднані лише з обмеженою кількістю нейронів попереднього шару, переважно розташованих на периферії локальної області. Таке розташування забезпечує ефективну обробку інформації та вилучення відповідних ознак. В архітектурі ЗНМ ваги зберігаються постійними в кожній площині, що призводить до, так званих, згорткових шарів. Після згорткових шарів є шар зменшення розмірності, який досягає цього зменшення через процес локального усереднення. Цей процес далі повторюється з додатковими згортковими шарами, що призводить до ієрархічної організації всередині мережі. У міру того, як архітектура мережі просувається через свої рівні, вона стає здатною отримувати більш узагальнені функції, на які менше впливають спотворення зображення. Ця здатність особливо корисна для класифікаційних завдань. Під час навчання ЗНМ використовується стандартний метод зворотного поширення для виправлення помилок, як описано в [10]. У порівнянні з традиційною БНМ, ЗНМ демонструє явні переваги як у швидкості, так і в

надійності для цілей класифікації. Одним із помітних атрибутів ЗНМ є те, що ознаки, створені на вищих рівнях мережі, можна ефективно використовувати для класифікації за допомогою таких методів, як підхід «найближчого сусіда», який передбачає обчислення евклідової відстані. Крім того, ЗНМ здатні ефективно вивчати функції навіть для зображень, які не були включені в навчальний набір.

ЗНМ демонструє вражаючу швидкість з точки зору як навчання, так і продуктивності, що робить її дуже перспективною архітектурою для майбутніх досягнень у сфері розпізнавання просторових об’єктів. Крім того, слід зазначити, що будь-яка навчена модель ЗНМ має здатність визначення ймовірності того, що зображення належать до класів, до яких вони віднесені.

1.2.3 Метод головних компонентів

Метод головних компонентів (МГК) – це метод для аналізу даних, які складаються з кількох кількісних залежних змінних, що взаємопов’язані між собою [11].

У контексті розпізнавання обличчя даний метод, в основному, використовується для представлення зображення людини за допомогою компактного вектора, відомого як вектор головних компонентів. Далі даний вектор порівнюється з еталонними векторами, що зберігаються в базі даних. Основна мета методу головних компонентів полягає в ефективному зменшенні розмірності простору ознак, гарантуючи, що він охоплює найбільш репрезентативні ознаки, спільні для всіх індивідів. При застосуванні даного методу стає можливим ідентифікувати та описувати різні джерела мінливості, які присутні в навчальному наборі зображень обличчя. Після ідентифікації джерел мінливості, вони описуються за допомогою набору ортогональних векторів, відомих як власні вектори.

У завданні розпізнавання обличчя початкова колекція зображень перетворюється в єдину уніфіковану матрицю даних. Кожен рядок у матриці представляє певне зображення особи, розташоване в послідовному порядку.

Для забезпечення одноманітності в межах навчального набору зображення всіх особистостей, обличчя яких містяться в базі даних, змінюються до однакових розмірів, а їхні гістограми нормалізуються. Як правило, стовпці матриці X відцентровані і середнє значення кожного стовпця дорівнює нулю. Крім того, коли змінні всередині матриці вимірюються з використанням різних одиниць, прийнято стандартизувати кожну змінну, щоб мати норму одиниці. У МГК компоненти таблиці даних X можуть бути отримані за допомогою процесу декомпозиції сингулярного значення (ДСЗ).

Декомпозиція за сингулярним значенням є більш узагальненою формою декомпозиції, яку можна застосувати для ідентифікації відповідних компонентів. Власні компоненти матриці охоплюють як власні вектори, так і власні значення, які є числовими значеннями та векторами, пов'язаними з квадратними матрицями. Ці компоненти спільно сприяють правильному розкладу матриці, що допомагає аналізувати базову структуру матриці.

Варто зазначити, що не всі квадратні матриці мають правильне розкладання. Однак для матриць, таких як кореляційні або коваріаційні, правильне розкладання приймає відносно просту форму. Належне розкладання цих типів матриць має важливе значення, оскільки воно використовується для визначення максимальних або мінімальних значень функцій, які їх описують. Концепція декомпозиції сингулярного значення передбачає розбиття прямокутної матриці на три різні матриці: дві ортогональні матриці та одну діагональну матрицю.

Якщо розглянемо прямокутну матрицю A з розмірами n × m, де кількість рядків (n) ≥ кількості стовпців (m), то представлення матриці, що є результатом, можна виразити наступним чином:

$$B = U\Gamma V^T, \tag{1.1}$$

де U – матриця розмірами n × m, в якій стовпці U є ортонормованими, задовольняють умові $U^TU = I$. Дані стовпці називаються лівими сингулярними векторами матриці B; V – матриця, яка є ортонормальною матрицею m × m, задовольняє умові $V^TV = I$. Стовпці V відомі як праві сингулярні вектори матриці B; D - матриця, яка є діагональною, що складається з елементів, які є або нульовими, або додатними і називаються сингулярними значеннями.

Основним результатом розкладання сингулярних значень матриці зображення X у контексті розпізнавання обличчя є поява набору власних векторів, відомих як власні грані обличчя, які візуально представлено на рис. 1.4 [12].

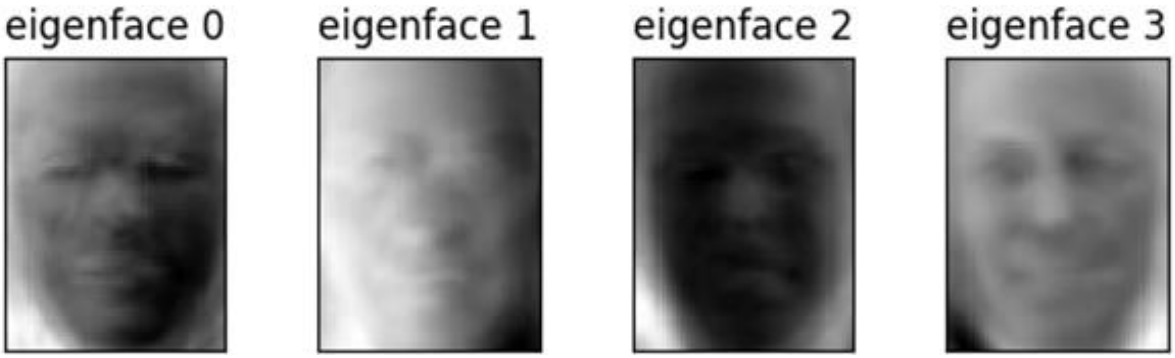

Рисунок 1.4 – Набір з власних векторів (власні обличчя)

Набір чітких характеристик обличчя, відомих як головні компоненти, можна отримати шляхом множення вектора вихідного зображення на вектор, що представляє обличчя людини, для кожного власного вектора. Для того, щоб відновити вихідне зображення, необхідно додати власні вектори, які були помножені на набір рис обличчя людини.

Приклад побудови обличчя людини з використанням гібриду власних облич (Eigenfaces) та головних компонент приведено на рис. 1.5. У деяких випадках бажано отримати лише найбільш релевантну інформацію з матриці

даних. У таких ситуаціях завдання полягає у визначенні оптимальної кількості головних компонент для розгляду.

Один з підходів полягає в побудові графіку власних значень компонент у порядку їх зменшення та спостеріганні точки, де нахил графіка різко змінюється – від крутого до пологого. Дана точка, відома як "точка зламу", характеризує кількість компонент, що зберігають найбільшу кількість інформації про дані [14].

Рисунок 1.5 – Приклад побудови обличчя людини з використанням гібриду власних облич (Eigenfaces) та головних компонент

1.3 Огляд існуючих систем біометричного розпізнавання

Для проведення порівняльного аналізу існуючих систем біометричного розпізнавання обрано наступні:

– Amazon Rekognition Service (Cloud-середовище);

– Microsoft Azure Face API;

– FacePhi (біометрична автентифікація).

1.3.1 Microsoft Azure Face API

Microsoft Azure Face API – це частина хмарних послуг Microsoft Azure, яка представляє собою передову технологію розпізнавання облич. Ця система надає можливість інтегрувати функції розпізнавання облич у програми та послуги, використовуючи алгоритми машинного навчання та штучного інтелекту. Azure Face API здатна виконувати кілька ключових функцій, яка може ідентифікувати людину на зображенні, порівнюючи її обличчя з базою даних відомих облич, що робить її корисною для систем безпеки та персоналізованих користувацьких взаємодій. Крім того, система здатна аналізувати риси обличчя та визначаючи такі атрибути, як вік, стать, емоції, а також наявність окулярів чи бороди. Однією з особливостей Azure Face API є її гнучкість і масштабованість, що дозволяє інтегруватися в різні застосунки та сервіси. Система також постійно вдосконалюється, щоб забезпечити високу точність і надійність у розпізнаванні облич. Система широко використовується у різноманітних галузях, включаючи безпеку, маркетинг, персоналізоване обслуговування клієнтів та соціальні дослідження клієнтів.

1.3.2 Amazon Rekognition Service

Amazon Rekognition Service є частиною хмарних сервісів Amazon Web Services (AWS) і представляє собою передову систему аналізу зображень та відео. Вона використовує глибоке навчання для розпізнавання об'єктів, облич, тексту та, навіть, для визначення сцен та активності в переданих медіафайлах. Основною особливістю Amazon Rekognition є її здатність до розпізнавання облич, що дозволяє не лише ідентифікувати індивідуальних людей на зображеннях та у відео, але й аналізувати їх вирази обличчя, щоб визначити емоції, наприклад радість чи смуток. Система також може виявляти демографічні характеристики, такі як вік чи стать. Крім того, Amazon Rekognition надає можливості по виявленню незвичайної активності у відео, що робить її корисною для систем відеоспостереження та безпеки. Також, система широко використовується у різних галузях, включаючи

маркетинг, аналітику соціальних медіа та розробку користувацького інтерфейсу.

Важливою особливістю є її масштабованість та інтеграція з іншими сервісами AWS, що забезпечує гнучкість у використанні для великих компаній та індивідуальних розробників. Amazon Rekognition постійно оновлюється для покращення точності та розширення функціональних можливостей.

На рис. 1.6 представлений приклад використання Amazon Rekognition з іншими сервісами для розробки системи розпізнавання обличчя [13].

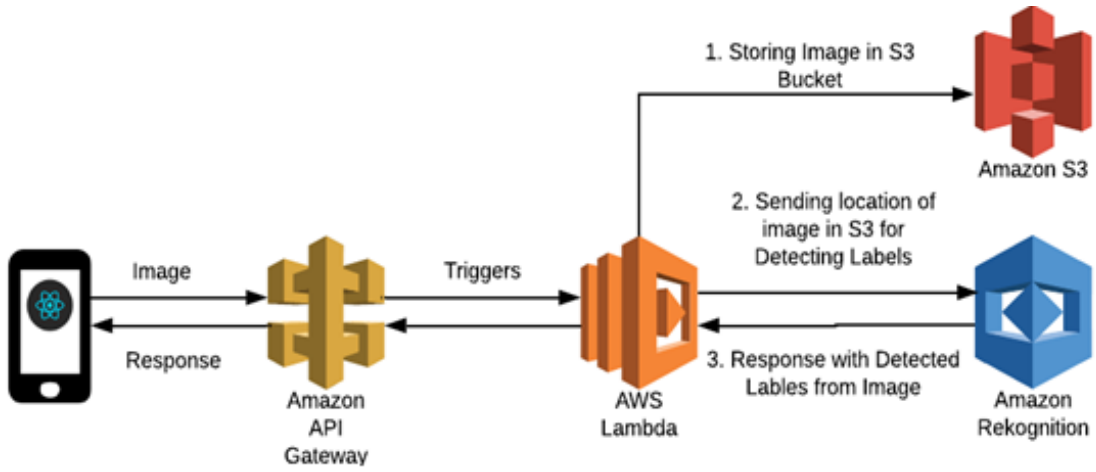

Рисунок 1.6 – Архітектура системи з використанням Amazon Rekognition

Користувач завантажує зображення через мобільний застосунок, після чого Amazon API Gateway отримує зображення обличчя та передає його в AWS Lambda, яка виконує наступні дії. На першому етапі зображення зберігається в Amazon S3, хмарному сховищі, а далі AWS Lambda передає місце зберігання зображення в Amazon Rekognition для його аналізу.

Сервіс Rekognition розпізнає обличчя або інші об'єкти на зображенні та повертає результати у вигляді міток назад в AWS Lambda. Далі мітки надсилаються до користувача через API Gateway, і мобільний застосунок отримує відповідь з результатами розпізнавання.

1.3.3 FacePhi

FacePhi – це відома компанія, яка спеціалізується на вдосконаленні та наданні передових технологій розпізнавання обличчя та рішень біометричної автентифікації та займається розробкою програмного пакету системи біометричної автентифікації FacePhi, який має широкий спектр пропозицій та може легко інтегруватися з web-застосунками та іншими системами, забезпечуючи безпомилковий контроль доступу та протоколи індивідуальної ідентифікації. FacePhi забезпечує максимальну точність ідентифікації користувачів, тим самим підвищуючи загальні заходи безпеки. Комплексний пакет FacePhi включає розширене рішення біометричної автентифікації на основі персональних даних. Ця система дозволяє користувачам легко отримувати доступ до систем або послуг, використовуючи унікальні характеристики обличчя для підтвердження своєї особи.

На рис. 1.7 представлена концепція розпізнавання обличчя з використанням ключових точок, яка є основою системи FacePhi для ідентифікації [15].

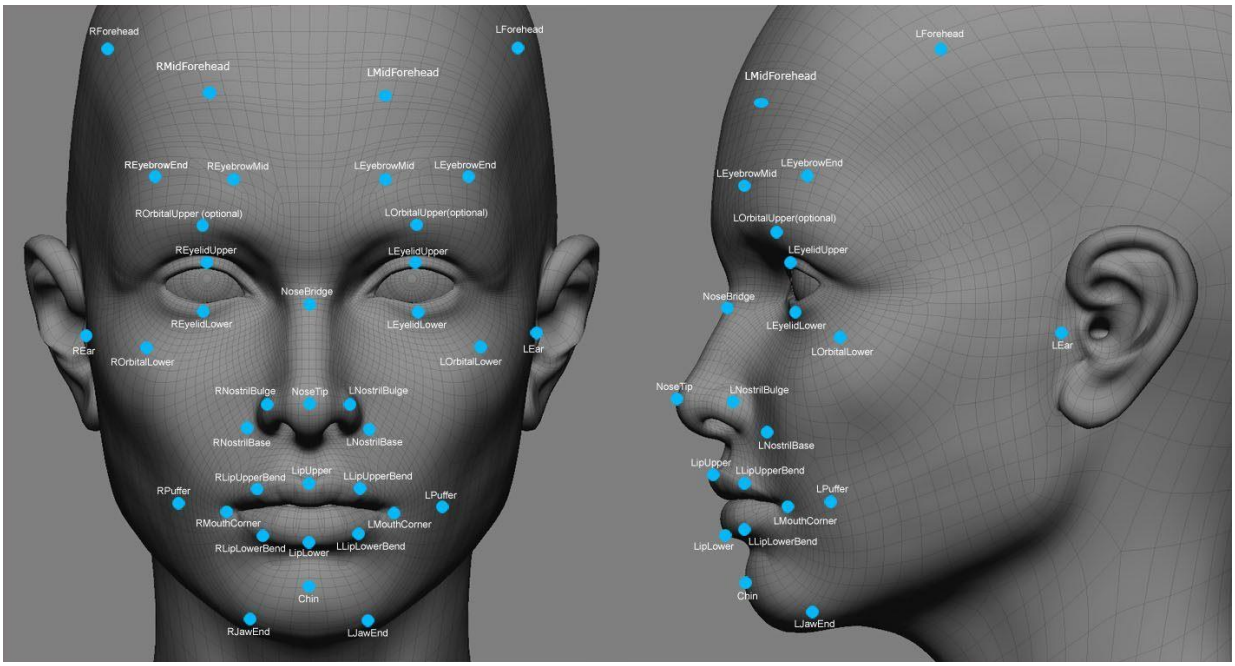

Рисунок 1.7 – Розпізнавання обличчя з використанням ключових точок

Концепція розпізнавання обличчя з використанням ключових точок полягає в тому, що система визначає специфічні анатомічні особливості обличчя, які є унікальними для кожної людини. Ключові точки обличчя можуть включати положення очей, носа, рота та інших характерних рис обличчя.

За допомогою аналізу розташування точок та відстаней між ними, система створює унікальний математичний шаблон або "відбиток" обличчя, який може використовуватися для точного розпізнавання та ідентифікації людини. Такий підхід дозволяє системам біометричної автентифікації, як FacePhi, забезпечувати високий рівень точності, незалежно від зовнішніх факторів.

1.4 Мета та постановка завдання

Метою кваліфікаційної роботи є розробка системи розпізнавання біометричних даних обличчя з використанням технологій машинного навчання.

Об’єктом дослідження є системи біометричного розпізнавання обличчя, предметом - моделі та методи ідентифікації обличчя користувача за біометричними характеристиками.

Для досягнення поставленої мети визначено наступні завдання:

– огляд методів та сучасних систем біометричного розпізнавання обличчя;

– аналіз відомих алгоритмічних рішень для біометричного розпізнавання обличчя;

– розробка системи розпізнавання біометричних даних обличчя з використанням сучасних технологій машинного навчання та нейронних мереж;

– аналіз та вибір технологій, інструментів та бібліотек для розробки системи;

– розробка алгоритму біометричного розпізнавання обличчя;

– розробка програмного забезпечення системи;

– тестування розробленої системи розпізнавання біометричних даних обличчя.

2 СУЧАСНІ АЛГОРИТМІЧНІ РІШЕННЯ ДЛЯ БІОМЕТРИЧНОГО РОЗПІЗНАВАННЯ ОБЛИЧЧЯ

У розділі розглянуто та проаналізовано сучасні алгоритми, що використовуються в системах біометричного розпізнавання облич, зокрема, алгоритм локальних бінарних шаблонів, алгоритми глибокого навчання та алгоритм Віоли-Джонса.

2.1 Алгоритм локальних бінарних шаблонів

Локальний бінарний шаблон (LBP) виділяється як надзвичайно ефективний оператор текстури, що призначає мітки пікселям зображення, виконуючи порогову операцію для околиць пікселя та інтерпретуючи результат як двійкове число. LBP використовує чотири параметри для виконання своїх операцій. Перший параметр означає радіус, що оточує центральний піксель, і відіграє вирішальну роль у побудові круглого локального бінарного шаблону. Як правило, його значення дорівнює 1. Другий параметр 2 визначає кількість точок вибірки, які використовуються для побудови кругового локального бінарного шаблону. Параметри 3 і 4 визначають кількість комірок у горизонтальному та вертикальному напрямках відповідно. Більша кількість комірок призводить до дрібнішої сітки та подальшого збільшення розмірності вектора, що є результатом ознак. Початковий процес обчислення гістограми передбачає створення проміжного зображення, яке ефективно фіксує визначальні риси вихідного зображення, особливо зосереджуючись на атрибутах обличчя.

Це досягається шляхом реалізації підходу ковзного вікна, який керується заздалегідь визначеними параметрами радіуса та сусідів пікселя, що використовуються для побудови локального бінарного шаблону.

Процес застосування оператора LBP представлено на рис. 2.1.

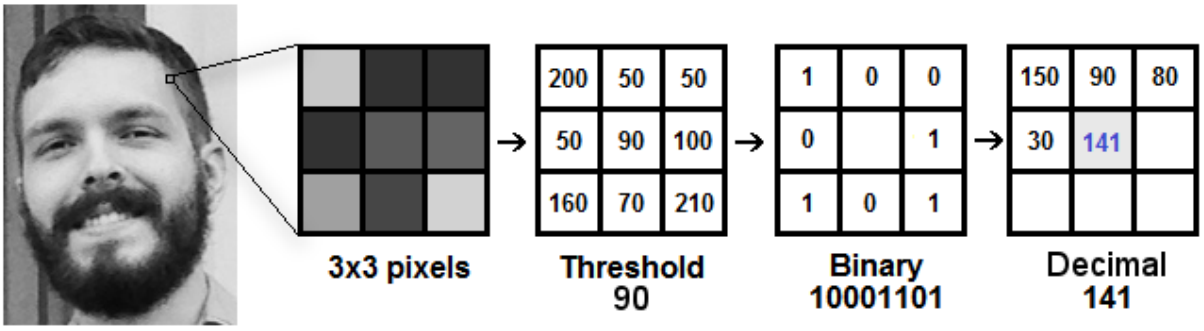

Рисунок 2.1 – Покроковий процес застосування оператора LBP

У методі локальних бінарних шаблонів, застосованому до зображень обличчя у відтінках сірого, із зображення виділяється невелике вікно, зазвичай 3x3 пікселі. Це вікно представлено у вигляді матриці 3x3, де кожен елемент вказує на інтенсивність пікселя в діапазоні від 0 до 255.

Центральне значення матриці вибрано як порогове значення. Потім даний поріг використовується для перетворення сусідніх пікселів у двійкові значення. Кожному з восьми сусідніх пікселів призначається двійкове значення: 1, якщо інтенсивність пікселя дорівнює або перевищує порогове значення, і 0, якщо інтенсивність пікселя нижче порогового значення. Значення центрального пікселя не перетворюється на двійкове значення.

Згодом двійкові значення сусідніх пікселів об’єднуються в одне двійкове число, що часто робиться шляхом зчитування значень рядка за рядком. Далі двійкове число перетворюється на десяткове число. Нарешті, десяткове число присвоюється центральному пікселю вихідної матриці 3x3.

Повторення цього процесу для всього зображення перетворює його на зображення, де піксель представляє локальні шаблони навколишньої області.

На рис. 2.2 представлено оператори, що відповідають деяким графічним примітивам (зліва направо – п’ятно/фон, п’ятно, кінець лінії, грань, кут) [14].

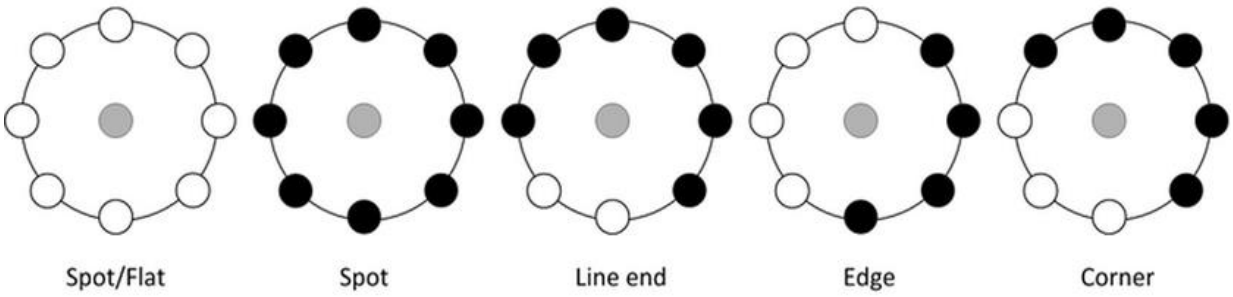

Рисунок 2.2 – Оператори, що відповідають графічним примітивам

Використовуючи оператор локальних бінарних шаблонів для кожного окремого пікселя зображення, в результаті буде сформована гістограма.

Отримана гістограма складається з окремих стовпців, у яких кожен стовпець представляє уніфікований код. Крім того, є додатковий стовпець, спеціально призначений для отримання інформації про будь-які неправильні шаблони, які можуть бути присутніми на зображенні. Нижче представлено опис роботи алгоритму локальних бінарних шаблонів [15].

Формально виразити опис оператора LBP можна наступним чином:

$$\mathrm{LBP}(x_c y_c) = \sum_{p=0}^{p-1} 2^P S(i_p - i_c), \tag{2.1}$$

де $x_c y_c$ – це координати центрального пікселя в зображенні, навколо якого буде виконуватися операція обчислення LBP; $i_p - i_c$ – інтенсивність пікселя, P – кількість пікселів в околі центрального пікселя (визначає кількість сусідів, які будуть використані для обчислення); S – функція порівняння, що виконує бінаризацію, яка виражена рівністю:

$$s(x) = \begin{cases} 1, \text{якщо } x \geq 0 \\ 0 \text{ інакше} \end{cases}, \tag{2.2}$$

Функція (2.2) повертає 1, якщо різниця між інтенсивністю сусіднього пікселя i_p та пікселя i_c більше або дорівнює 0; 0, якщо різниця менша за 0.

2.2 Алгоритми глибокого навчання

Глибоке навчання стало потужною технікою класифікації зображень, особливо в області виявлення та розпізнавання облич. У порівнянні з традиційними підходами до комп'ютерного бачення, глибоке навчання, зокрема за допомогою застосування згорткових нейронних мереж або ConvNet, незмінно демонструє вищу продуктивність. Примітно, що будь-яка нейронна мережа, яка включає принаймні один згортковий шар, може бути класифікована як CNN, що робить її невід'ємним компонентом цієї передової методології. На додаток до своїх можливостей у задачах розпізнавання обличчя, CNN знаходять широке застосування в різних системах комп'ютерного зору, включно з тими, які використовуються в робототехніці та автономних транспортних засобах, таких як безпілотні автомобілі.

Локальність є фундаментальним поняттям, яке має велике значення з точки зору людини. Коли людина дивиться на навколишній світ, очі сприймають об'єкти в невеликих, локалізованих областях зору та ретельно аналізують їхні чіткі особливості, такі як кути, краї та текстури. Дану концепцію також можна спостерігати в області згорткових нейронних мереж. Згорткові шари у таких мережах використовують фільтри, які проходять через вхідне зображення, вилучаючи локальні особливості, такі як межі та форми. Вони відіграють ключову роль у вилученні низькорівневих ознак, що формують основу для більш складних патернів на подальших шарах, таких як шари підсумовування та згладжування, що дозволяє мережі поступово отримувати глибші розуміння зображення.

Мережа LeNet-5, запропонована у 1989 році [16], була однією з перших згорткових нейронних мереж (CNN). Вона отримала визнання за свою здатність точно розпізнавати поштові індекси та номери. Надалі з'явилося

кілька більш досконалих архітектур CNN, які перевершують можливості LeNet. Однак фундаментальні принципи, що лежать в основі цих мереж, залишилися в основному незмінними з моменту заснування першої CNN.

На рис. 2.3 представлена архітектура згорткової нейронної мережі LeNet. Мережа призначена для визначення категорії, до якої належить вхідне зображення. Архітектура складається з чотирьох ключових операцій: згортки (Convolution), застосування функції ReLU, об'єднання та класифікації.

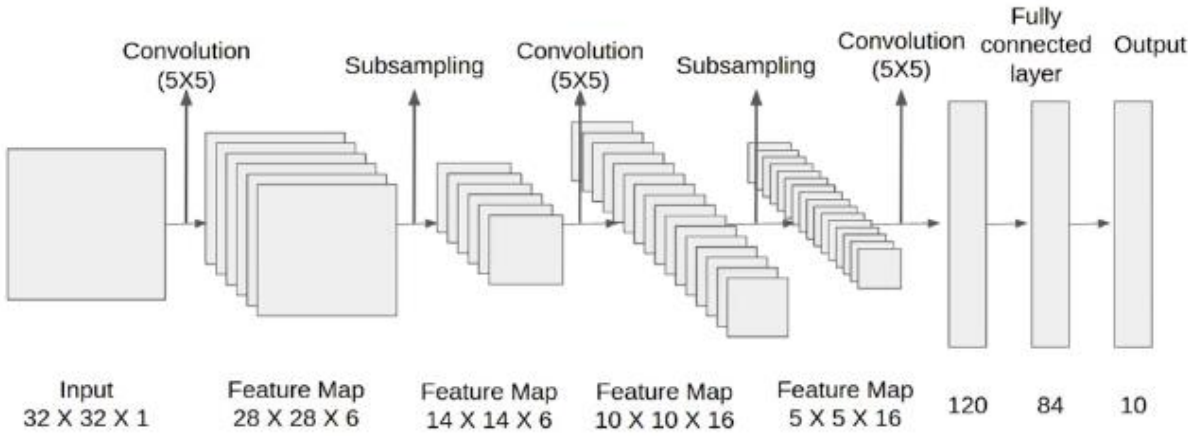

Рисунок 2.3 – Архітектура згорткової мережі LeNet-5

Згортка застосовується до вхідних даних та відіграє вирішальну роль у виділенні ознак із зображення. При використанні фільтрів основна мета згортки полягає у вивченні та ідентифікації відмінних атрибутів у вхідному зображенні. Слід зазначити, що згортка ефективно зберігає просторову кореляцію між пікселями під час цього процесу.

Для кращого розуміння розглянемо зображення в градаціях сірого, яке може бути адекватно представлене однією матрицею. У більшості випадків цього достатньо для таких завдань, як розпізнавання обличчя.

Тепер розглянемо матрицю зображення розміром 5 x 5 і фільтр або детектор функцій розміром 3 x 3. Коли даний фільтр застосовано, у матриці вибирається певна квадратна область 3 x 3. Шляхом множення відповідних

елементів матриці та детектора ознак добутки підсумовуються та записуються в матрицю 3 х 3, що є результатом, відомим як карта ознак.

Приклад карти ознак та фільтра представлений на рис. 2.4.

При аналізі матриці виконується процес відбору квадратів матриці, починаючи з самого першого елемента на рисунку зліва. Процес відбору можна представити як застосування фільтра до матриці. Після вибору кожного квадрата обчислюється добуток його елементів і додається до загальної суми елементів карти ознак. Згодом детектор ознак пересувається праворуч на одну клітинку, дозволяючи розглянути наступний квадрат. Якщо оброблено весь рядок, детектор повертається до початкової позиції та переміщується на одну клітинку вниз, забезпечуючи продовження операцій множення та додавання.

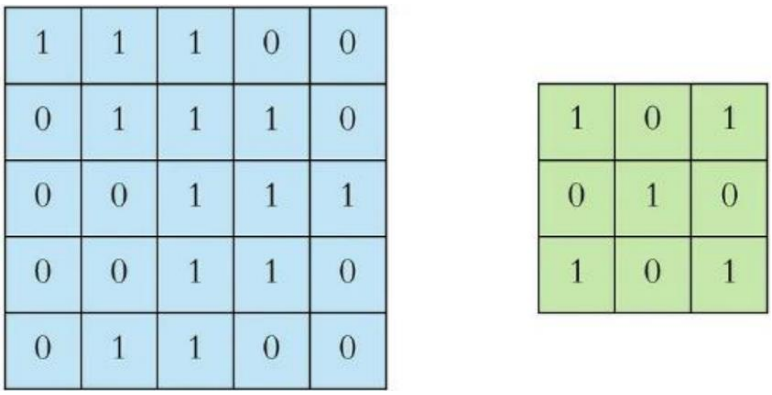

Рисунок 2.4 – Матриця зображення (5 х 5) та згортковий фільтр (3 х 3)

Конволюційний фільтр є невеликою матрицею фіксованих розмірів, зазвичай 3х3 або 5х5, яка використовується для обробки вхідного зображення в процесі згорткових операцій. Його основна функція полягає у вилученні специфічних ознак або патернів зображення, таких як краї, текстури або інші локальні особливості. Під час операції згортки фільтр переміщується по всій матриці пікселів зображення (вхідних даних) і для кожного положення

виконується операція елементного множення між елементами фільтра і відповідними пікселями зображення, після чого результати підсумовуються.

Це дозволяє конволюційному шару мережі виділяти найбільш релевантні характеристики, що згодом стають основою для подальших шарів глибокої нейронної мережі. У результаті кожен фільтр відповідає за певний тип ознаки, що дозволяє побудувати багаторівневу структуру аналізу зображень.

На рис. 2.5 представлений результат виконання операції двовимірної згортки застосування конволюційного фільтра 3x3.

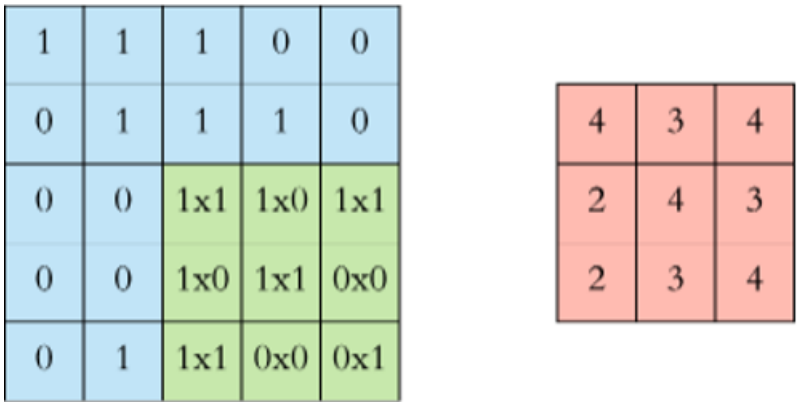

Рисунок 2.5 – Фінальна ітерація застосування конволюційного фільтра

Однак, у практичних застосуваннях згортки виконуються в 3D. Важливо взяти до уваги, що насправді зображення представляється як тривимірна матриця, яка включає розміри висоти, ширини та глибини.

У контексті представлення зображення компонент глибини відповідає різним кольоровим каналам, таким як червоний, зелений і синій (RGB). Отже, згортковий фільтр має певні параметри висоти та ширини, наприклад 3x3 або 5x5, і має на меті охопити всю глибину свого входу. Застосування фільтрів із різними значеннями до зображення призводить до різноманітних карт функцій. Фільтри служать для виконання таких операцій із зображенням, як виявлення країв, підвищення різкості, розмиття тощо.

На рис. 2.6 представлено застосування різноманітних детекторів ознак до одного зображення [14].

Operation	Kernel	Image result
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	

Рисунок 2.6 – Застосування різноманітних фільтрів до зображення

У сфері згорткових нейронних мереж значення фільтрів вивчаються самостійно під час процесу навчання, незважаючи на вимогу ручного налаштування певних параметрів. Параметри можуть включати кількість фільтрів, розміри фільтрів, архітектуру мережі та інші. Збільшуючи кількість фільтрів, згорткова мережа здатна виявляти більший набір ознак, що дозволяє досягати кращих результатів у задачі розпізнавання образів.

На розміри карти ознак впливають три ключові параметри, які необхідно визначити перед виконанням процесу згортки. По-перше, параметр глибини стосується кількості фільтрів, які використовуються під час згортки. Важливо зауважити, що кількість карт функцій відповідає

кількості використовуваних фільтрів. По-друге, параметр кроку визначає кількість пікселів, на яку переміщується фільтр по вихідному зображенню. Чим більший крок, тим меншою буде результуюча карта функцій. Нарешті, концепція заповнення нулями виникає в певних сценаріях, де виявляється вигідним заповнити матрицю вхідного зображення нулями. Це полегшує застосування фільтрів до пікселів, розташованих на межах зображення. Після виконання операції згортання застосовується функція ReLU, також відома як функція випрямленої лінійної одиниці або функція випрямлення, як продемонстровано на рис. 2.7.

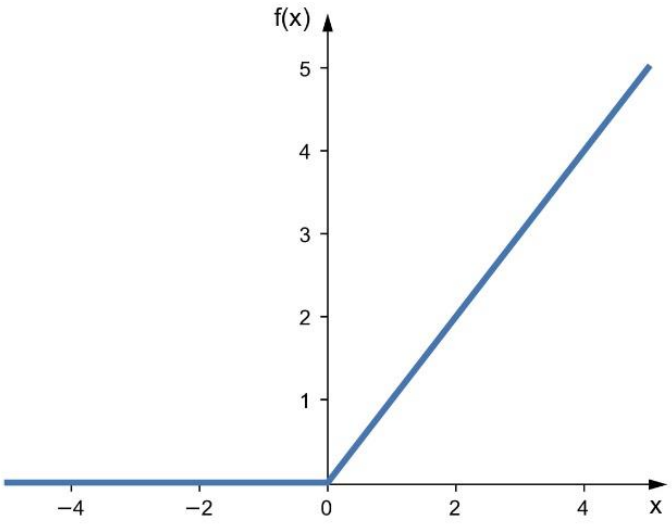

Рисунок 2.7 - Графік передавальної функції ReLU

Функція ReLU є функцією активації та визначається наступним чином:

$$f(x) = \max(0, x), \tag{2.3}$$

Функція ReLU використовується на поелементній основі, особливо коли йдеться про зображення, де вона працює на рівні пікселів. Отже, будь-які негативні значення пікселів на карті функцій замінюються нулями. Метою використання функції ReLU у згортковій нейронній мережі є

впровадження нелінійності. Це важливо, оскільки більшість реальних даних, які використовуються для навчання нейронної мережі, мають нелінійні характеристики. На етапі згортки, який включає поелементне множення та додавання матриць, виконуються лише лінійні операції.

Однак для врахування нелінійності включена функція ReLU.

Варто зазначити, що ReLU можна замінити альтернативними нелінійними функціями, як-от сигмоїда або гіперболічний тангенс, але в більшості випадків було доведено, що функція ReLU дає кращі результати [16].

Після виконання операції згортки зазвичай застосовують пулінг, щоб зменшити розмірність вхідних даних. Операція пулінгу служить для зменшення загальної кількості параметрів, що призводить до скорочення часу навчання та пом’якшення проблеми переобладнання. Шари пулінгу виконують зменшення дискретизації незалежно на кожній карті ознак, зменшуючи висоту та ширину, зберігаючи при цьому глибину незмінною. Серед різних типів методів об’єднання широко використовується максимальний пулінг, який передбачає вибір максимального значення в межах визначеного вікна. На відміну від операції згортки, пулінг не містить параметрів, натомість, пулінг-шар переміщується вікном по вхідних даних і отримує максимальне значення в кожному вікні.

На рис. 2.8 представлений приклад, в якому використовується вікно розміром 2x2, і обирається максимальне значення серед елементів, що знаходяться у вікні. Таким чином, з кожного вікна розміром 2x2 вибирається найбільше значення, яке стає частиною вихідної матриці після операції максимального пулінгу. У контексті застосування операції об’єднання, вікно плавно зміщується як горизонтально (вздовж осі x), так і вертикально (вздовж осі y) з невеликими кроками. Цей рух можна порівняти з поступовим рухом під час прокручування. На рис. 2.8 кожен крок уздовж осей x і ординат відповідає зміщенню двох комірок. Ще одна схожість з операцією згортки полягає в зменшенні розмірності карти ознак. На рисунку матриця до

об’єднання мала розміри 4 x 4. Однак після застосування операції об’єднання розміри було зменшено до 2 x 2. Важливо зазначити, що об’єднання виконується окремо для кожної окремої карти об’єктів.

Дана операція має важливе значення в згорткових мережах, оскільки служить кільком цілям.

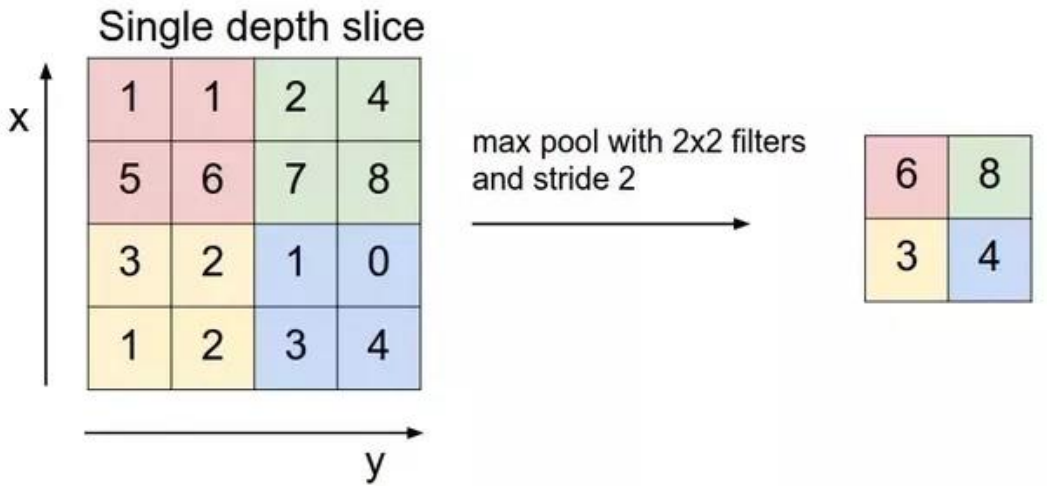

Рисунок 2.8 – Застосування операції об’єднання максимального значення

Однією з основних цілей є ефективне зменшення кількості параметрів і обчислень, необхідних у мережі. Таким чином, цей процес дає змогу краще контролювати можливість перетренування, здійснювати сценарій, коли модель демонструє високу точність під час класифікації даних навчання, але є нижчою, коли вводить нові, невидимі дані. Крім того, об'єднання також сприяє здатності згорткової мережі залишатися відносно вільним від незначних спотворень і рухів, присутніх у вхідному зображенні. Завдяки операції об’єднання (pooling) мережа досягає рівня нечутливості до цих незначних змін, що підвищує її надійність.

Мережа LeNet структурована наступним чином. Спочатку виконується ряд операцій згортки, використання функції ReLU та операції об’єднання. Далі названі операції повторюються з використанням результатів попереднього шару. Кількість повторень залежить від архітектури згорткової нейронної мережі.

У сучасних згорткових нейронних мережах може бути кілька рівнів згортки, ReLU та об’єднання, іноді мають десятки шарів. Також можна послідовно виконувати операції згортки та ReLU перед застосуванням об’єднання. Це дозволяє мережі розпізнавати дедалі складніші функції. Наприклад, коли згортка вперше застосована до вхідного зображення, вона виявляє краї.

Подальші операції згортання визначають прості форми, які надалі використовуються для виявлення складніших функцій, таких як форма обличчя. Глибина та складність можливостей розпізнавання мережі зростає з кількістю виконуваних операцій згортки.

У сфері нейронних мереж операції згортання та об’єднання дають значні функції, які далі передаються на повністю зв’язаний рівень (fully-connected), також відомий як багатошаровий перцептрон. Даний тип нейронної мережі має три основні рівні: вхідний, прихований і вихідний рівні, позбавлені будь-яких петель зворотного зв’язку.

У повністю зв’язаному шарі кожен нейрон попереднього шару складно з’єднаний з кожним нейроном наступного шару. Використовуючи функцію активації Softmax, а також інші активаційні функції, перцептрон визначає клас, до якого належить вхідне зображення. Це визначення залежить від застосування «знань», набутих під час фази навчання.

Для ілюстрації розглянемо сучасну архітектуру згорткової нейронної мережі, відому як ResNet. ResNet є багаторівневою глибокою мережею, побудованою на основі залишкових з'єднань (skip connections). Цей підхід дозволяє ефективно мінімізувати проблему затухання градієнтів у процесі навчання, яка традиційно ускладнює оптимізацію параметрів глибоких моделей.

Ключовим технічним досягненням ResNet стала її здатність забезпечувати рівень точності класифікації зображень, який порівнюється із когнітивними можливостями людини. Така точність обумовлена здатністю архітектури адаптуватися до складних структурних характеристик

зображень, таких як багатовимірна варіативність об’єктів, неоднорідність текстур, зміщення та масштабна інваріантність.

Точність класифікації у контексті ResNet формалізується через метрики, що враховують не лише правильність розпізнавання об’єктів, але й узгодженість моделі при роботі із зашумленими даними та викривленнями. Саме це зробило ResNet базовим еталоном для оцінки продуктивності у галузі комп’ютерного зору та дозволило суттєво підвищити ефективність розв’язання завдань аналізу візуальної інформації. До появи ResNet дослідники намагалися покращити можливості згорткових нейронних мереж шляхом збільшення їх глибини. Однак збільшення глибини часто призводило до того, що продуктивність мережі не тільки не покращувалася, але й погіршувалася через проблему зникнення градієнта, що ускладнювало процес навчання.

Результату погіршення продуктивності мережі сприяють два основні фактори: по-перше, реалізація ідентичних відображень стає складною на наступних рівнях, а додавання зайвих рівнів може негативно вплинути на функціональність мережі та процес навчання. По-друге, зі збільшенням глибини нейронної мережі виникають труднощі в обчисленні градієнтів для зворотного поширення помилок. Існування функцій активації призводить або до експоненціального збільшення, або до майже нульового зменшення значень градієнта всередині мережі, що ще більше ускладнює процес навчання. Для вирішення проблем, згаданих вище, ResNet використовує модулі Residual замість звичайних шарів. Модулі являють собою складені шари, які вводять контури, полегшуючи встановлення ідентичного відображення та сприяючи поширенню градієнтів помилок у протилежному напрямку під час навчання.

Рис. 2.9 ілюструє архітектуру нейронної мережі ResNet50. На рисунку зверху приведено модуль Residual, який демонструє приклад мережі з 50 шарами, а на рисунку нижче зображено послідовний Residual-модуль.

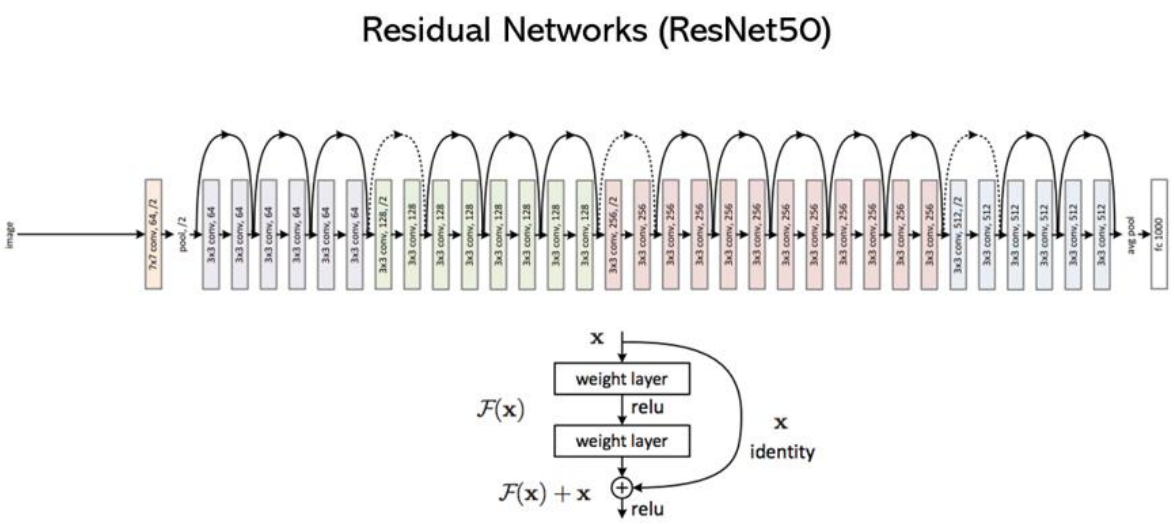

Рисунок 2.9 – Архітектура нейронної мережі ResNet50

На відміну від звичайних мереж, нейронна мережа ResNet може досягати значної глибини, зазвичай перевищуючи 18 шарів і потенційно досягаючи до 1024 шарів. Із глибиною 152 шарів мережа досягає рівня помилок лише 4,49%.

2.3 Алгоритм Віоли-Джонса

Метод Віоли-Джонса, запропонований П. Віолою та М. Джонсом у 2001 році, є алгоритмом детекції облич, який базується на швидкому обчисленні простих ознак на основі інтегральних зображень, використанні каскаду слабких класифікаторів, побудованих через алгоритм AdaBoost, і структурі каскадної класифікації для швидкої фільтрації областей зображення. Основу методу складає використання Haar-подібних ознак, які представляють собою відносні різниці яскравості між прямокутними областями зображення.

Формально, ознака h обчислюється як:

$$h = \sum_{(x,y)\in R_1} I(x,y) - \sum_{(x,y)\in R_2} I(x,y), \quad (2.4)$$

де I(x,y) – інтенсивність пікселя в інтегральному зображенні, а R_1 та R_2 – прямокутні області, для яких розраховується різниця.

Інтегральне зображення S(x,y) необхідне для швидкого обчислення таких ознак, визначається як:

$$S(x,y) = \sum_{x' \le x, y' \le y} I(x', y'), \tag{2.5}$$

На основі (2.4) сума пікселів у будь-якому прямокутнику зображення за сталий час O(1) може бути обчислена за формулою:

$$Sum_R = S(x_2, y_2) - S(x_1, y_2) - S(x_2, y_1) + S(x_1, y_1), \tag{2.6}$$

де (x_1, y_2) і (x_2, y_2) – координати протилежних вершин прямокутника.

Для вибору найбільш значущих ознак застосовується алгоритм AdaBoost, який побудований на принципі перетворення набору слабких класифікаторів $h_i(x)$ у сильний класифікатор H(x). Слабкий класифікатор вибирається для кожної ознаки з урахуванням ваг помилок:

$$h_i(x) = \begin{cases} 1, \text{якщо } p_i f_i(x) < p_i \theta_i \\ 0, \text{інакше} \end{cases}, \tag{2.7}$$

де θ – загальний поріг класифікації.

Для того, щоб підвищити ефективність алгоритму, класифікатори організовані у вигляді каскаду: кожен рівень каскаду поступово відсіює області, які з високою ймовірністю не містять облич, і лише перспективні області передаються на наступний рівень.

Формально, кожен класифікатор C_k на рівні k приймає рішення:

$$C_k(x) = \begin{cases} \text{reject, якщо } H_k(x) < T_k \\ \text{pass, інакше} \end{cases}, \quad (2.8)$$

де H_k – сильний класифікатор рівня k, а T_k – відповідний поріг.

Важливою характеристикою методу Віоли-Джонса є його здатність працювати з різними масштабами об'єктів. Завдяки масштабуванню вікон аналізу алгоритм виконує багатомасштабний пошук, що дозволяє детектувати обличчя незалежно від їхніх розмірів або позицій на зображенні. Водночас цей підхід вимагає високої оптимізації, оскільки великий обсяг пошуку може значно збільшити обчислювальні витрати. Додатково, метод передбачає використання стратегій для зменшення кількості хибнопозитивних результатів, таких як коригування порогів класифікації на кожному рівні каскаду, підвищення ваги ключових ознак у процесі навчання та включення даних негативного зразка в процес побудови класифікаторів. Такий підхід забезпечує високу надійність моделі навіть в умовах сильної зашумленості чи варіативності вхідних зображень. Завдяки своїй архітектурі та теоретичній основі, метод Віоли-Джонса є основоположним у задачах детекції об'єктів та залишається актуальним, навіть незважаючи на появу більш складних моделей на основі глибоких нейронних мереж. Його перевагою є висока швидкість обробки, що дозволяє використовувати його у пристроях із обмеженими ресурсами, наприклад, у вбудованих системах чи мобільних додатках.

3 РОЗРОБКА СИСТЕМИ РОЗПІЗНАВАННЯ БІОМЕТРИЧНИХ ДАНИХ ОБЛИЧЧЯ

У даному розділі розроблено алгоритм біометричного розпізнавання обличчя, дано опис використаних технологій, інструментів та бібліотек для розробки системи, зокрема мови програмування, інструментів та бібліотек, здійснено опис розробленого коду з лістингами роботи програми.

3.1 Опис використаних технологій

Для розробки системи розпізнавання біометричних даних обличчя були використані наступні технології: мова програмування Python, а також бібліотеки OpenCV, Tensorflow, DeepFace, Numpy, Streamlit та Pandas.

3.1.1 Мова програмування Python

Python є високорівневою інтерпретованою мовою програмування, яка здобула популярність завдяки своїй простоті та зручності використання. Завдяки читабельному синтаксису Python дозволяє розробникам легко писати й підтримувати код, що робить його ідеальним вибором для швидкої розробки та прототипування. Однією з ключових особливостей цієї мови є її динамічна типізація, що означає, що типи змінних визначаються під час виконання, забезпечуючи велику гнучкість у написанні коду. Python активно використовується для розробки різноманітних програмних рішень, зокрема у сфері машинного навчання та обробки зображень, що є важливим у контексті розпізнавання біометричних даних обличчя. Завдяки багатій стандартній бібліотеці та підтримці численних сторонніх пакетів, Python дозволяє легко реалізовувати складні алгоритми та працювати з великими обсягами даних.

Для створення системи розпізнавання біометричних даних обличчя на Python можуть бути використані бібліотеки, такі як OpenCV для обробки зображень, та TensorFlow або PyTorch для побудови моделей глибокого навчання. OpenCV забезпечує інструменти для захоплення й аналізу зображень, а TensorFlow і PyTorch дозволяють створювати та тренувати нейронні мережі для класифікації та розпізнавання облич. Python також підтримує інтеграцію з іншими платформами, що дозволяє розробляти багатофункціональні системи з використанням різноманітних технологій.

3.1.2 Бібліотека OpenCV

OpenCV - скорочення від Open Source Computer Vision Library, служить комплексним інструментом для комп’ютерного зору та обробки зображень, яка надає різноманітний набір алгоритмів і функцій для дослідження візуальної інформації, що обслуговує не лише двовимірну, але й тривимірну обробку зображень, дозволяючи виконувати такі складні завдання, як виявлення об’єктів, вирівнювання камери та навіть стереоскопія. Важливим аспектом OpenCV є його надзвичайна ефективність, яка досягається завдяки точно налаштованим алгоритмам, закодованим на C/C++ і використанню інструкцій SIMD і методів апаратного прискорення, як-от використання GPU. Це робить OpenCV особливо корисною для систем, які потребують обробки зображень у реальному часі з мінімальною затримкою. Бібліотека OpenCV забезпечує інструменти для обробки та аналізу зображень, такі як перетворення геометричних фігур, сегментація та виділення контурів, фільтрація та згладжування, а також складніші алгоритми, такі як гістограма напрямлених градієнтів та розпізнавання за допомогою каскадів Хаара.

Існуючі інструменти для обробки зображень дозволяють створювати інтелектуальні системи, що здатні ідентифікувати та класифікувати об'єкти в зображеннях та відео. Крім того, OpenCV підтримує інтеграцію з іншими бібліотеками машинного навчання, такими як TensorFlow та PyTorch, що дозволяє розробникам використовувати глибокі нейронні мережі для

підвищення точності та надійності алгоритмів розпізнавання. Це відкриває можливості для реалізації складних моделей, таких як глибокі конволюційні нейронні мережі, які можуть бути навчені на великих наборах даних для виконання задач, що вимагають високого рівня розуміння візуальної інформації. Завдяки своїй гнучкості та широким можливостям, OpenCV широко використовується в різних сферах, включаючи робототехніку, доповнену реальність, автоматизовані системи контролю якості та системи безпеки. Це робить OpenCV незамінним інструментом для розробки рішень, які покладаються на обробку зображень та комп'ютерний зір.

3.1.3 Бібліотека Tensorflow

TensorFlow – це високопродуктивна, відкрита платформа для машинного навчання, розроблена компанією Google, яка забезпечує потужні інструменти для створення, тренування та розгортання моделей глибокого навчання. Завдяки своєму масштабованому дизайну, TensorFlow дозволяє розробникам працювати з моделями різної складності, від простих логістичних регресій до глибоких нейронних мереж, що використовуються у задачах комп'ютерного зору, обробки природної мови, та інших областях штучного інтелекту. Однією з ключових характеристик TensorFlow є його здатність виконувати обчислення на різних платформах, включаючи CPU, GPU та TPU, що дозволяє ефективно використовувати апаратні ресурси для прискорення тренування моделей.

TensorFlow надає широкий спектр високорівневих API, таких як Keras, які спрощують процес створення моделей завдяки інтуїтивно зрозумілому інтерфейсу та абстракціям. Також TensorFlow підтримує розподілені обчислення, що дозволяє тренувати моделі на кластерах серверів, розподіляючи обчислювальні задачі між кількома пристроями.

3.1.4 Бібліотека DeepFace

DeepFace – це популярна відкрита бібліотека для розпізнавання обличчя, яка базується на сучасних технологіях глибокого навчання. Вона забезпечує високу точність і надійність при виконанні різноманітних задач, пов'язаних із біометричним аналізом обличчя, таких як верифікація особистості, розпізнавання, класифікація за віком та статтю, а також визначення емоційного стану.

Однією з ключових особливостей DeepFace є її здатність інтегруватися з різними передовими моделями глибокого навчання, включаючи VGG-Face, Google FaceNet, OpenFace, Facebook DeepFace та інші. Це дозволяє користувачам вибирати найбільш підходящу модель для конкретного завдання або порівнювати результати кількох моделей для досягнення оптимальної точності. DeepFace підтримує як попередньо навчені моделі, так і можливість їх додаткового тренування на власних наборах даних, що дозволяє адаптувати рішення під специфічні потреби проєкту.

Бібліотека також забезпечує високу продуктивність, дозволяючи виконувати розпізнавання обличчя в реальному часі, що є критично важливим для додатків, які вимагають негайного реагування, таких як системи безпеки або інтерактивні користувацькі інтерфейси. Бібліотека DeepFace розроблена з урахуванням зручності використання, пропонуючи інтуїтивний API, який дозволяє розробникам легко інтегрувати функції розпізнавання обличчя у свої програми.

DeepFace включає в себе широкий спектр функцій для роботи з зображеннями, таких як виявлення облич, передобробка зображень, вирівнювання обличчя, нормалізація, а також побудова та порівняння векторних ознак (embedding), які використовуються для ідентифікації та верифікації. DeepFace також підтримує обробку зображень як на центральних процесорах, так і на графічних процесорах, що забезпечує гнучкість у виборі апаратного забезпечення в залежності від вимог продуктивності та ресурсів.

3.1.5 Бібліотека Numpy

NumPy – це базова бібліотека для наукових обчислень у Python, яка надає потужні інструменти для роботи з багатовимірними масивами та матрицями, а також широкий набір математичних функцій для виконання складних обчислень. Завдяки своєму високопродуктивному ядру, написаному на мові C, NumPy забезпечує значно вищу швидкість обробки даних у порівнянні зі стандартними засобами Python, що робить її незамінною для завдань, пов'язаних із чисельними розрахунками.

Однією з ключових можливостей NumPy є підтримка багатовимірних масивів (ndarray), які дозволяють ефективно зберігати та обробляти великі обсяги даних. Ці масиви можуть мати будь-яку кількість вимірів, що робить NumPy гнучким інструментом для роботи з даними будь-якої структури, від простих одномірних векторів до складних багатовимірних тензорів, які часто використовуються в машинному навчанні та обробці зображень.

NumPy також надає широкий набір функцій для роботи з масивами, включаючи операції над елементами, алгебру матриць, лінійну алгебру, статистичні операції та генерацію випадкових чисел. Ці функції оптимізовані для роботи з великими наборами даних і підтримують векторизацію операцій, що дозволяє зменшити час обчислень за рахунок паралельної обробки даних.

3.1.6 Бібліотека Streamlit

Streamlit – це сучасний фреймворк для створення інтерактивних веб-застосунків, орієнтований на розробників і науковців, які працюють з даними. Завдяки простоті та гнучкості, Streamlit дозволяє швидко створювати прототипи та розгортати аналітичні інструменти, візуалізації та моделі машинного навчання без необхідності глибоких знань у веб-розробці.

Однією з головних переваг Streamlit є його інтуїтивний API, що дозволяє перетворювати скрипти Python на повноцінні веб-застосунки всього за кілька рядків коду. Streamlit автоматично генерує інтерфейси для введення даних, графіків, таблиць та інших компонентів, що дозволяє розробникам

зосередитися на логіці додатка, не турбуючись про створення та налаштування користувацького інтерфейсу.

Streamlit підтримує інтерактивність, що дозволяє користувачам взаємодіяти із застосунком у режимі реального часу, змінюючи параметри, вводячи дані та миттєво отримуючи оновлені результати. Це досягається за рахунок автоматичного повторного виконання скрипту при зміні будь-якого входу, що забезпечує динамічне оновлення контенту додатка. Фреймворк Streamlit також інтегрується з популярними бібліотеками для роботи з даними, такими як Pandas, NumPy, Matplotlib, Plotly та Seaborn, що дозволяє легко включати візуалізації, графіки та діаграми в застосунок. Streamlit підтримує роботу з моделями машинного навчання, що дозволяє демонструвати та тестувати моделі безпосередньо в браузері, надаючи інтерфейс для налаштування параметрів і виведення результатів [19].

3.1.7 Бібліотека Pandas

Pandas – це потужна та універсальна бібліотека для роботи з даними в Python, яка забезпечує високорівневі структури даних та інструменти для ефективної маніпуляції, аналізу та візуалізації табличних даних. Вона особливо корисна для обробки великих обсягів даних, які можуть бути організовані у вигляді серій або DataFrame, що є двовимірною таблицею, схожою на електронну таблицю з лініями та стовпцями.

Однією з ключових можливостей Pandas є її здатність обробляти дані з різних джерел, включаючи файли CSV, Excel, SQL-бази даних та інші формати. Завдяки потужному механізму імпорту та експорту, Pandas дозволяє легко інтегрувати дані з різних джерел в єдину аналітичну платформу, що спрощує підготовку даних для подальшого аналізу або візуалізації.

Pandas надає широкий спектр інструментів для роботи з даними, включаючи фільтрацію, групування, агрегацію, об'єднання та злиття таблиць, а також складні операції, такі як обробка відсутніх значень, маніпуляція з

часовими рядами та створення складних умовних вибірок. Ці функції дозволяють ефективно очищувати та трансформувати дані, підготувавши їх до більш глибокого аналізу або моделювання. Однією з ключових особливостей Pandas є її здатність виконувати векторизовані операції над даними, що дозволяє значно прискорити обробку великих наборів даних. Завдяки використанню оптимізованих алгоритмів і внутрішніх обчислювальних структур, Pandas забезпечує високу продуктивність, дозволяючи швидко виконувати складні розрахунки та аналітичні завдання.

3.2 Опис програмної реалізації системи розпізнавання

На рисунку 3.1 представлена блок-схема алгоритму, який використовується для вилучення біометричних характеристик облич.

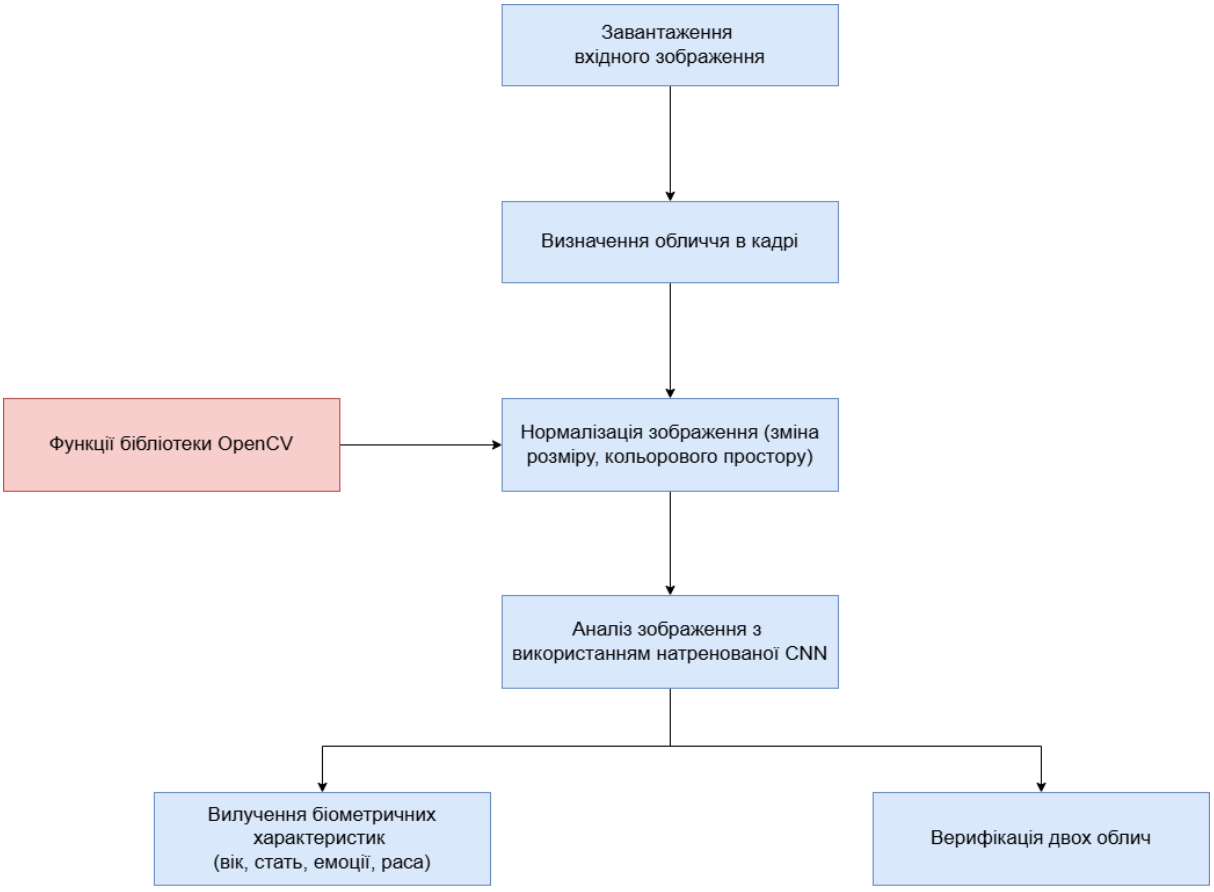

Рисунок 3.1 – Блок-схема алгоритму для вилучення біометричних характеристик облич

У відповідності до блок-схеми алгоритму процес розпізнавання обличчя починається з завантаження вхідного зображення, після чого виконується визначення обличчя у кадрі. Далі зображення піддається нормалізації, що включає зміну його розміру та кольорового простору за допомогою бібліотеки OpenCV. Після нормалізації зображення передається для аналізу за допомогою натренованої згорткової нейронної для екстракції ознак з обличчя, представленого на зображенні.

Далі можливі два варіанти: перший – це вилучення біометричних характеристик, таких як вік, стать, емоції та раса; другий – верифікація двох облич, тобто перевірка їх відповідності до різних зображень.

Головними функціями системи є функції, що безпосередньо виконують основні завдання, пов'язані з розпізнаванням облич.

До головних функцій системи можна віднести: виявлення обличя в кадрі, розпізнавання обличчя, зіставлення декількох облич та відображення текстових анотацій.

До допоміжних функцій можна віднести перетворення в массив, передача, декодування зображення в необхідний формат для подальшої обробки, перетворення закодованиого зображення у формат RGB.

У лістингу 3.1 представлена реалізація допоміжних функцій розробленої системи розпізнавання, які будуть використані при розробці головних функцій програмної системи.

Функція init_data призначена для ініціалізації даних шляхом завантаження їх з файлу або створення нового DataFrame, якщо такий файл не існує. Вона приймає три параметри: data_path, що визначає шлях до файлу, cols_info, що містить список колонок для основної інформації, і cols_encode, що додає колонки для закодованих даних. Функція спочатку перевіряє, чи існує файл за вказаним шляхом за допомогою

os.path.isfile(data_path). Якщо файл існує, функція завантажує його у вигляді DataFrame за допомогою pd.read_csv(data_path). Якщо файл не знайдено, функція створює порожній DataFrame з колонками, визначеними об'єднанням списків cols_info і cols_encode. Функція byte to array використовує np.frombuffer, щоб перетворити байтовий потік у numpy-масив, і передає його до OpenCV функції cv2.imdecode, яка здійснює декодування зображення в формат, придатний для обробки. Функція bgr_to_rgb призначена для перетворення зображення, закодованого у форматі BGR (який за замовчуванням використовує OpenCV), у формат RGB, що є більш загальноприйнятим для роботи з зображеннями в інших бібліотеках та додатках.

Вона отримує на вхід масив зображення у форматі BGR і використовує функцію cv2.cvtColor, щоб конвертувати його у формат RGB.

Лістинг 3.1 – Реалізація допоміжних функцій розробленої системи розпізнавання

```
def init_data(data_path: str, cols_info: List, cols_encode: List):
    if os.path.isfile(data_path):
        return pd.read_csv(data_path)
    else:
        return pd.DataFrame(columns=cols_info + cols_encode)

def byte_to_array(image_in_byte):
    return cv2.imdecode(
        np.frombuffer(image_in_byte.read(), np.uint8),
        cv2.IMREAD_COLOR
    )

def bgr_to_rgb(image_in_array):
    return cv2.cvtColor(image_in_array, cv2.COLOR_BGR2RGB)

def face_distance_to_conf(face_distance, face_match_threshold: float = 0.6):
    if face_distance > face_match_threshold:
        range = (1.0 - face_match_threshold)
        linear_val = (1.0 - face_distance) / (range * 2.0)
        return linear_val
    else:
        range = face_match_threshold
        linear_val = 1.0 - (face_distance / (range * 2.0))
```

```
        return linear_val + ((1.0 - linear_val) * np.power((linear_val - 0.5) * 2, 0.2))
```

Функція bgr_to_rgb призначена для перетворення зображення, закодованого у форматі BGR (який за замовчуванням використовує OpenCV), у формат RGB, що є більш загальноприйнятим для роботи з зображеннями в інших бібліотеках та додатках. Вона отримує на вхід масив зображення у форматі BGR і використовує функцію cv2.cvtColor, щоб конвертувати його у формат RGB. Функція face_distance_to_conf призначена для перетворення дистанції обличчя на коефіцієнт впевненості (confidence score), який визначає ймовірність того, що два обличчя відповідають одному і тому ж індивіду. Вона приймає на вхід значення дистанції обличчя (face_distance) і порогове значення для відповідності обличчя (face_match_threshold, яке за замовчуванням дорівнює 0.6). Якщо дистанція перевищує порогове значення, функція розраховує linear_val, використовуючи формулу, яка поступово зменшує впевненість залежно від відстані.

У лістингу 3.2 представлена реалізація класів, які призначені для організації та зберігання даних у структурованому вигляді.

Лістинг 3.2 – Реалізація класів для організації та зберігання даних

```
class Detection(SimpleNamespace):
    bbox: List[List[float]] = None
    landmarks: List[List[float]] = None
class Identity(SimpleNamespace):
    detection: Detection = Detection()
    name: str = None
    embedding: np.ndarray = None
    face: np.ndarray = None
class Match(SimpleNamespace):
    subject_id: Identity = Identity()
    gallery_id: Identity = Identity()
    distance: float = None
```

Клас Detection призначений для зберігання даних про детекцію обличчя та має два атрибути:

– bbox, який є списком списків з числами з плаваючою комою і зберігає координати прямокутної рамки (bounding box), що визначає межі обличчя на зображенні;

– landmarks, який також є списком списків з числами з плаваючою комою і зберігає координати ключових точок обличчя, таких як очі, ніс і рот.

Клас Identity призначений для зберігання інформації про ідентифіковану особу. Він має кілька атрибутів:

– detection, який є об'єктом класу Detection і зберігає дані про детекцію обличчя;

– name, який є рядком (str) і містить ім'я особи;

– embedding, який є масивом NumPy (np.ndarray) і зберігає вектор ознак обличчя, що використовується для ідентифікації;

– face, який також є масивом NumPy і містить саме зображення обличчя.

Клас Match призначений для зберігання інформації про зіставлення (матчинг) двох осіб. Він має кілька атрибутів:

– subject_id і gallery_id, які є об'єктами Identity і представляють особу, яку ідентифікують, та особу в галереї зображень;

– distance, який є числом з плаваючою комою і містить відстань між двома векторами ознак двох осіб, що використовується для визначення схожості.

У додатку Б в лістингу 3.3 приведено програмний код, який реалізує секцію розпізнавання біометричних атрибутів обличчя.

Наведений код реалізує частину користувацького інтерфейсу на основі бібліотеки Streamlit для створення веб-застосунків.

Робота системи починається з перевірки авторизації користувача: чи авторизований користувач, використовуючи змінну LOGGED_IN.

Якщо користувач авторизований, у бічній панелі (sidebar) відображається заголовок "Settings" за допомогою методу st.sidebar.title.

Далі, за допомогою методу st.sidebar.selectbox, користувачу пропонується вибрати тип аналізу з меню, яке містить чотири опції: "Verification", "Recognition", "Recognition from Picture" і "Recognition from Webcam". Вибір зберігається у змінній user_choice.

Якщо користувач обрав опцію "Recognition", у центральній частині сторінки відображається заголовок "Face Recognition Section" за допомогою методу st.title. Після цього користувачу пропонується завантажити зображення для аналізу, використовуючи метод st.file_uploader. Завантажені файли можуть бути у форматах "jpg", "png", або "jpeg". Після завантаження зображення, користувачу пропонується вибрати додаткові опції для аналізу обличчя. Після завантаження зображення, користувачу пропонується вибрати додаткові опції для аналізу за допомогою бібліотеки DeepFace, використовуючи метод st.multiselect. Доступні опції включають: age, gender, emotion, race. Обрані опції зберігаються у змінній tasks для передачі в метод.

У додатку Б в лістингу 3.4 приведено програмний код, який реалізує секцію з біометричним розпізнаванням обличчя та можливістю збереження в БД.

На першому кроці визначаються кілька констант, які будуть використовуватись в процесі обробки. PATH_TO_DATA встановлює шлях до файлу з базою даних облич у форматі CSV.

COLOR_DARK та COLOR_WHITE задають кольори для малювання рамок і тексту на зображенні, а COLS_INFO та COLS_ENCODE задають структуру колонок у базі даних: інформаційні колонки для імен та описів, а також колонки для збереження векторів ознак облич (128 чисел). Далі, використовуючи st.title, відображається заголовок секції "Face Recognition From Picture". Користувачу пропонується завантажити зображення, яке містить обличчя, через метод st.file_uploader. Після завантаження зображення, якщо файл не є порожнім, зображення перетворюється у формат масиву за допомогою функції byte_to_array.

На наступному кроці за допомогою бібліотеки face_recognition, виконується пошук облич на зображенні (метод face_locations). Кожне знайдене обличчя обрамляється прямокутником (cv2.rectangle), а біля обрамлення додається індекс обличчя (cv2.putText). Зображення з накладеними обрамленнями далі відображається на сторінці Streamlit за допомогою st.image. Якщо на зображенні виявлено одне або більше облич, користувач може вибрати одне з них за допомогою st.selectbox, який пропонує вибір ідентифікатора обличчя.

Після цього база даних завантажується з CSV-файлу через функцію init_data, де містяться інформація про імена, описи, і кодування облич.

Обране обличчя перетворюється у вектор ознак за допомогою face_recognition.face_encodings. Далі розраховується дистанція між вектором ознак обраного обличчя і кожним обличчям з бази даних (метод face_recognition.face_distance).

Пораховані дистанції використовуються для оцінки схожості облич, і результати відображаються у вигляді таблиці з п’ятьма найближчими результатами за схожістю, відсортованими за зростанням дистанції. Користувачу також надається можливість додати нове обличчя до бази даних, якщо він активує чекбокс "Add new face to faces database". У цьому випадку користувачу пропонується ввести ім'я та опис обличчя. Якщо користувач натискає кнопку "Add", нове обличчя додається до бази даних, і вона зберігається у CSV-файлі. Після цього відображається повідомлення про успішне додавання обличчя.

Якщо на завантаженому зображенні не було виявлено жодного обличчя, користувачу відображається повідомлення про помилку.

Далі, у лістингу 3.5 представлена реалізація секції з біометричного розпізнавання обличчя в реальному часі з використанням потокового відео. Програмний код призначений для виявлення облич і обчислення їх ключових точок (landmarks) у відеокадрі або зображенні з використанням бібліотеки MediaPipe і модуля FaceMesh.

Створюється екземпляр детектора облич FACE_DETECTOR, який налаштовується на вдосконалення ключових точок обличчя (refine_landmarks=True), при цьому встановлюється обмеження на максимальну кількість виявлених облич (max_num_faces=7), а також встановлюються мінімальні пороги впевненості для виявлення та відстеження облич (min_detection_confidence=0.5, min_tracking_confidence=0.5).

Функція detect_faces, яка приймає кадр (зображення) у форматі np.ndarray і повертає список об'єктів типу Detection, відповідає за безпосереднє виявлення облич у кадрі.

На початку функції detect_faces викликається метод FACE_DETECTOR.process(frame), який обробляє кадр і повертає результат з інформацією про знайдені обличчя та їх ключові точки.

Лістинг 3.5 – Реалізація секції з біометричного розпізнавання обличчя в реальному часі з використанням потокового відео

```
SIMILARITY_THRESHOLD = 1.0
FACE_RECOGNIZER = rt.InferenceSession(
  "model.onnx",
  providers=rt.get_available_providers()
        )
  FACE_DETECTOR = mp.solutions.face_mesh.FaceMesh(
         refine_landmarks=True,
         max_num_faces=7,
         min_detection_confidence=0.5,
         min_tracking_confidence=0.5
        )
  def detect_faces(frame: np.ndarray) -> List[Detection]:
         result = FACE_DETECTOR.process(frame)
         detections = []
         if result.multi_face_landmarks:
  for               count,               detection               in
enumerate(result.multi_face_landmarks):
  five_landmarks  =  np.asarray(detection.landmark)[[470,  475,  1,
57, 287]]
        landmarks = np.asarray(
       [[landmark.x     *     frame.shape[1],     landmark.y     *
frame.shape[0]] for landmark in five_landmarks]
                  )
```

```
     all_x_coords = [landmark.x * frame.shape[1] for landmark in
detection.landmark]
    all_y_coords = [landmark.y * frame.shape[0] for landmark in
detection.landmark]
    x_min,         x_max         =         int(min(all_x_coords)),
int(max(all_x_coords))
    y_min,         y_max         =         int(min(all_y_coords)),
int(max(all_y_coords))
          bbox = [[x_min, y_min], [x_max, y_max]]
               detection = Detection(
               idx=count,
               bbox=bbox,
               landmarks=landmarks,
               confidence=None
                    )
               detections.append(detection)
               return detections
```

Якщо обличчя виявлені (if result.multi_face_landmarks:), функція проходить по кожному виявленому обличчю в результатах. За допомогою вбудованої функції enumerate, отримується індекс і деталі кожного обличчя. Для кожного обличчя здійснюється вибір п’яти специфічних ключових точок, які визначаються із загального списку ключових точок обличчя. Обрані точки перетворюються у numpy-масив і масштабуються відповідно до розмірів кадру. Далі виконується обчислення координат всіх ключових точок обличчя для побудови прямокутника (bounding box) навколо обличчя. Для цього з масиву координат обчислюються мінімальні та максимальні значення по осях X і Y, які визначають верхній лівий і нижній правий кути прямокутника. Далі створюється об'єкт типу Detection, який містить інформацію про індекс обличчя, координати bounding box, ключові точки обличчя і надалі об'єкт зберігається у список detections. Після проходження по всім виявленим обличчям, функція detect_faces повертає список detections, який містить детальну інформацію про всі знайдені обличчя в кадрі.

У лістингу 3.6 представлена реалізація функції, яка призначена для розпізнавання облич на основі вхідного кадру і списку виявлених облич.

Лістинг 3.6 – Реалізація функції, яка призначена для розпізнавання облич на основі вхідного кадру і списку виявлених облич

```
def recognize_faces(
            frame: np.ndarray,
            detections: List[Detection]
        ) -> List[Identity]:
            if not detections:
                return []
            identities = []
            for detection in detections:
                landmarks_target = np.array(
                    [
                        [38.2946, 51.6963],
                        [73.5318, 51.5014],
                        [56.0252, 71.7366],
                        [41.5493, 92.3655],
                        [70.7299, 92.2041],
                    ],
                    dtype=np.float32
                )
                tform = SimilarityTransform()
                tform.estimate(detection.landmarks,
landmarks_target)
                tmatrix = tform.params[0:2, :]
                face_aligned  =  cv2.warpAffine(frame,  tmatrix,
(112, 112), borderValue=0.0)
    input_image  =  (np.asarray([face_aligned]).astype(np.float32)
/ 255.0).clip(0.0, 1.0)
    embedding   =   FACE_RECOGNIZER.run(None,   {"input_image":
input_image})[0][0]
                identities.append(
                    Identity(
                        detection=detection,
                        embedding=embedding,
                        face=face_aligned
                    )
                )
            return identities
```

Функція recognize_faces приймає на вхід кадр у форматі np.ndarray та список об'єктів Detection, які містять інформацію про розташування і ключові точки виявлених облич. Якщо обличчя були виявлені, функція ініціалізує порожній список identities, куди будуть зберігатися результати розпізнавання. Далі, для кожного обличчя у списку detections, проводиться наступний процес:

1. Визначаються еталонні координати ключових точок обличчя landmarks_target, які використовуються для вирівнювання облич.

Координати відповідають стандартним позиціям ключових точок на обличчі.

2. Створюється об'єкт SimilarityTransform для обчислення матриці перетворення, яка буде використовуватися для вирівнювання обличчя. За допомогою методу estimate, матриця перетворення (tmatrix) обчислюється на основі відповідності між виявленими ключовими точками обличчя (detection.landmarks) і еталонними точками (landmarks_target).

3. Обличчя вирівнюється на основі обчисленої матриці перетворення за допомогою функції cv2.warpAffine, і результат зберігається в змінній face_aligned. Отримане вирівняне обличчя має розмір 112x112 пікселів.

4. Вирівняне обличчя конвертується у формат зображення для подальшої обробки: значення пікселів нормалізуються до діапазону [0, 1], і дане зображення передається до нейронної мережі для отримання вектора ознак обличчя (embedding). Вектор ознак обличчя може бути отримано шляхом виконання моделі розпізнавання облич FACE_RECOGNIZER.run, де input_image є нормалізованим вирівняним обличчям.

5. Створюється об'єкт Identity, який містить інформацію про розпізнане обличчя: об'єкт Detection, вектор ознак embedding, і зображення face_aligned.

У лістингу 3.7 представлено реалізацію функції match_faces, яка виконує порівняння векторів ознак облич із двох груп: суб'єктів (subjects) і галереї (gallery), щоб знайти найбільш схожі пари облич. Спершу перевіряється, чи містять обидва списки необхідні елементи; якщо хоча б один із них порожній, функція match_faces негайно повертає порожній список, оскільки порівнювати немає що. Далі створюються масиви векторів ознак для облич з галереї і суб'єктів, що здійснюється вилученням ознак кожного обличчя з об'єктів типу Identity. За допомогою методу cosine_distances обчислюються косинусні відстані між цими векторами ознак. Косинусна відстань використовується для вимірювання схожості між двома

векторами, при цьому, менша відстань вказує на більшу схожість. Після цього функція проходить через кожного суб'єкта в списку subjects, обчислюючи для нього відстані до всіх облич у галереї. Вибирається індекс обличчя з галереї, яке має найменшу косинусну відстань до даного суб'єкта. Якщо мінімальна відстань менша за певний поріг схожості (SIMILARITY_THRESHOLD), то пара облич вважається відповідністю (match), і відповідний об'єкт Match додається до списку matches. Об'єкт Match містить інформацію про суб'єкта, відповідне обличчя з галереї та їхню косинусну відстань.

Після того, як усі суб'єкти були порівняні, функція сортує список збігів за іменем облич у галереї, що дозволяє упорядкувати результати за певним критерієм.

Лістинг 3.7 – Реалізація функції, яка призначена для порівняння векторів ознак облич для знаходження найбільшого збігу

```
def match_faces(
            subjects: List[Identity],
            gallery: List[Identity]
        ) -> List[Match]:
            if len(gallery) == 0 or len(subjects) == 0:
                return []
            embs_gal    =    np.asarray([identity.embedding    for
identity in gallery])
            embs_det    =    np.asarray([identity.embedding    for
identity in subjects])
            cos_distances = cosine_distances(embs_det, embs_gal)
            matches = []
            for ident_idx, identity in enumerate(subjects):
                dists_to_identity = cos_distances[ident_idx]
                idx_min = np.argmin(dists_to_identity)
                if          dists_to_identity[idx_min]          <
SIMILARITY_THRESHOLD:
                    matches.append(
                        Match(
                            subject_id=identity,
                            gallery_id=gallery[idx_min],
                            distance=dists_to_identity[idx_min]
                        )
                    )
            matches    =    sorted(matches,    key=lambda    match:
match.gallery_id.name)
```

```
            return matches
```

У лістингу 3.8 приведено програмний код, який призначений для запуску потокового відео для для розпізнавання облич в реальному часі.

Лістинг 3.8 – Запуск потокового відео для розпізнавання облич в реальному часі

```
gallery = []
 for file in files:
 file_bytes = np.asarray(bytearray(file.read()), dtype=np.uint8)
 img  =  cv2.cvtColor(cv2.imdecode(file_bytes,  cv2.IMREAD_COLOR),
cv2.COLOR_BGR2RGB)
            detections = detect_faces(img)
            if detections:
                subjects = recognize_faces(img, detections[:1])
                gallery.append(
                    Identity(
                        name=os.path.splitext(file.name)[0],
                        embedding=subjects[0].embedding,
                        face=subjects[0].face
                    )
                )
        gal_container.image(
            image=[identity.face for identity in gallery],
            caption=[identity.name for identity in gallery]
        )
        st.markdown("**Live Stream**")
        with st.container():
            webrtc_streamer(
                key="LiveFaceRecognition",
                mode=WebRtcMode.SENDRECV,
                video_frame_callback=video_frame_callback,
                media_stream_constraints={
                    "video": {"width": 1280},
                    "audio": False}
    )
```

Код запуску потокового відео реалізує процес додавання облич до галереї, а також відображення зображень облич і запуску живого стріму для розпізнавання облич в реальному часі. При реалізації створюється порожній список gallery, який буде використовуватись для збереження об'єктів Identity, що представляють виявлені і розпізнані обличчя. Зображення конвертується з формату BGR у RGB для коректного відображення і обробки. Далі, функція

detect_faces використовується для виявлення облич на зображенні. Якщо обличчя були виявлені, функція recognize_faces виконує розпізнавання облич, обмежуючи аналіз лише першим знайденим обличчям. Отримані дані зберігаються у вигляді об'єкта Identity, який містить ім'я файлу зображення, вектор ознак обличчя та саме обличчя у вирівняному форматі. Цей об'єкт додається до списку gallery. Після цього, всі зображення з галереї обличчя відображаються на сторінці разом із підписами, що відповідають іменам файлів, з яких ці обличчя були взяті. Далі на сторінці відображається заголовок "Live Stream", який інформує користувача про те, що нижче надається потік живого відео. Далі створюється контейнер, в якому запускається відеострімінг з використанням технології Web Real-Time Communication (WebRTC). Для цього використовується функція webrtc_streamer, яка налаштовує стрімінг у режимі SENDRECV – одночасна передача і отримання відео.

Відеопотік обробляється через функцію зворотного виклику video_frame_callback, яка обробляє кожен кадр відео для розпізнавання облич.

4 ТЕСТУВАННЯ СИСТЕМИ РОЗПІЗНАВАННЯ БІОМЕТРИЧНИХ ДАНИХ ОБЛИЧЧЯ

При розробці системи розпізнавання біометричних даних обличчя було проведено тестування розробленої системи.

На рис. 4.1 представлена початкова сторінка, яка доступна при першому запуску системи, де користувач має можливість:

– пройти реєстрацію;

– пройти автентифікацію;

– відновити втрачений пароль до системи.

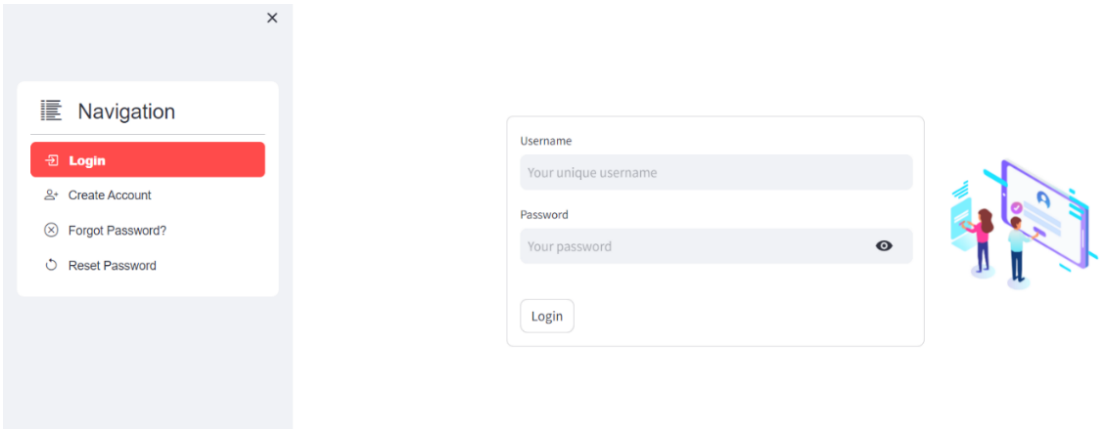

Рисунок 4.1 – Початкова сторінка системи для проходження автентифікації

На рис. 4.2 представлена сторінка, де користувач має можливість зареєструватися з наступними даними: ім’я, електронна пошта, пароль.

Після успішної авторизації в системі користувач отримує доступ до основних функцій.

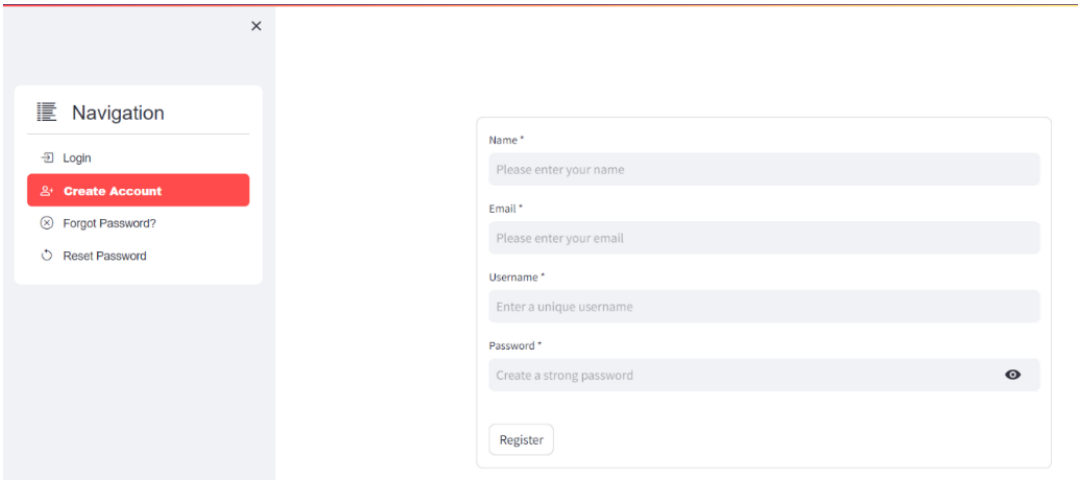

Рисунок 4.2 – Сторінка системи для проходження реєстрації

На рис. 4.3 представлена сторінка, де відбувається верифікація на основі біометричних параметрів обличчя.

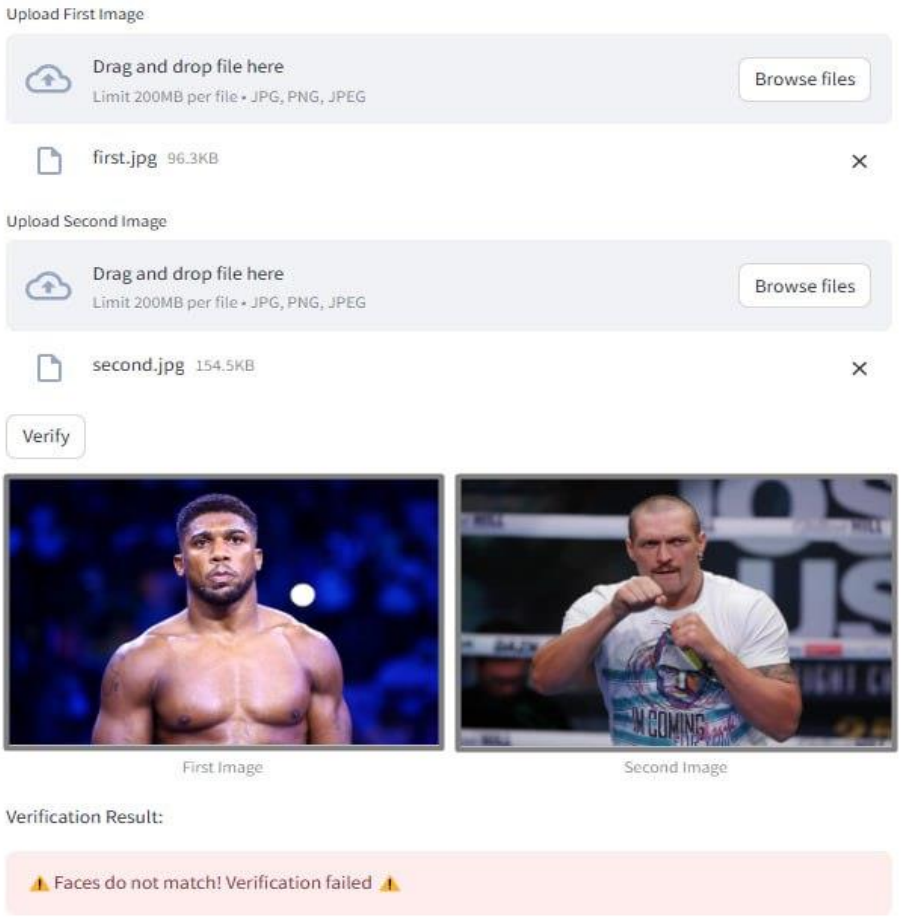

Рисунок 4.3 – Сторінка з верифікацією на основі біометричних параметрів обличчя

Як видно з рисунка 4.3, для проведення тестування було завантажено зображення різних особистостей і отримано результат про відсутність збігу.

На рис. 4.4 представлено тестування із зображеннями однієї особи. З рисунка видно, що верифікація успішно пройдена.

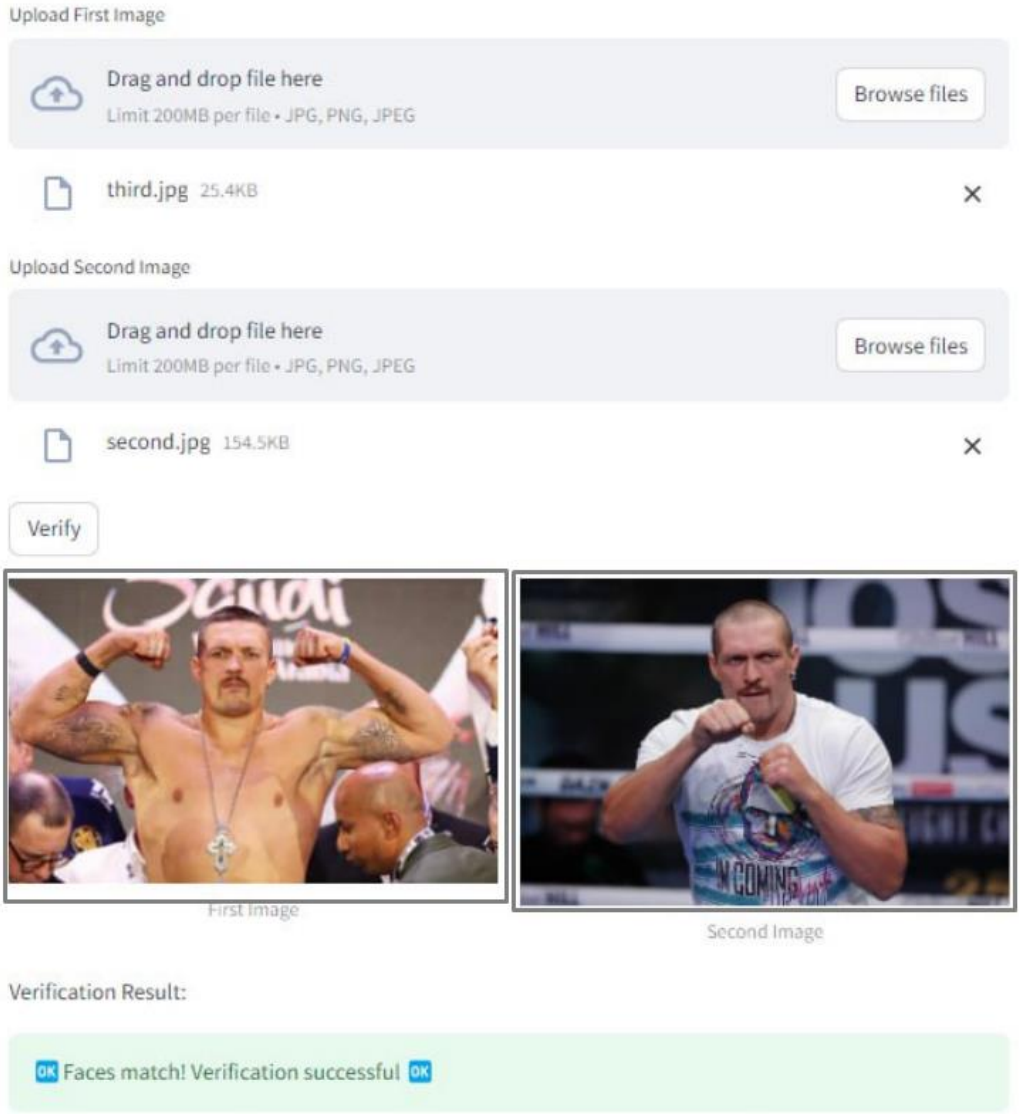

Рисунок 4.4 – Результат успішної верифікації особи

На рисунку 4.5 представлений інтерфейс, який здійснює аналіз обличчя за допомогою бібліотеки DeepFace. У верхній частині інтерфейсу користувачу пропонується вибрати опції для аналізу обличчя, зокрема вік (age), стать (gender), емоції (emotion) та расу (race). Всі опції обрані і відображені у вигляді червоних тегів під текстовим полем.

Під фотографією особи є кнопка "Analyze", яка запускає процес розпізнавання та аналізу обличчя за обраними критеріями.

Результати розпізнавання представлені у таблиці під зображенням.

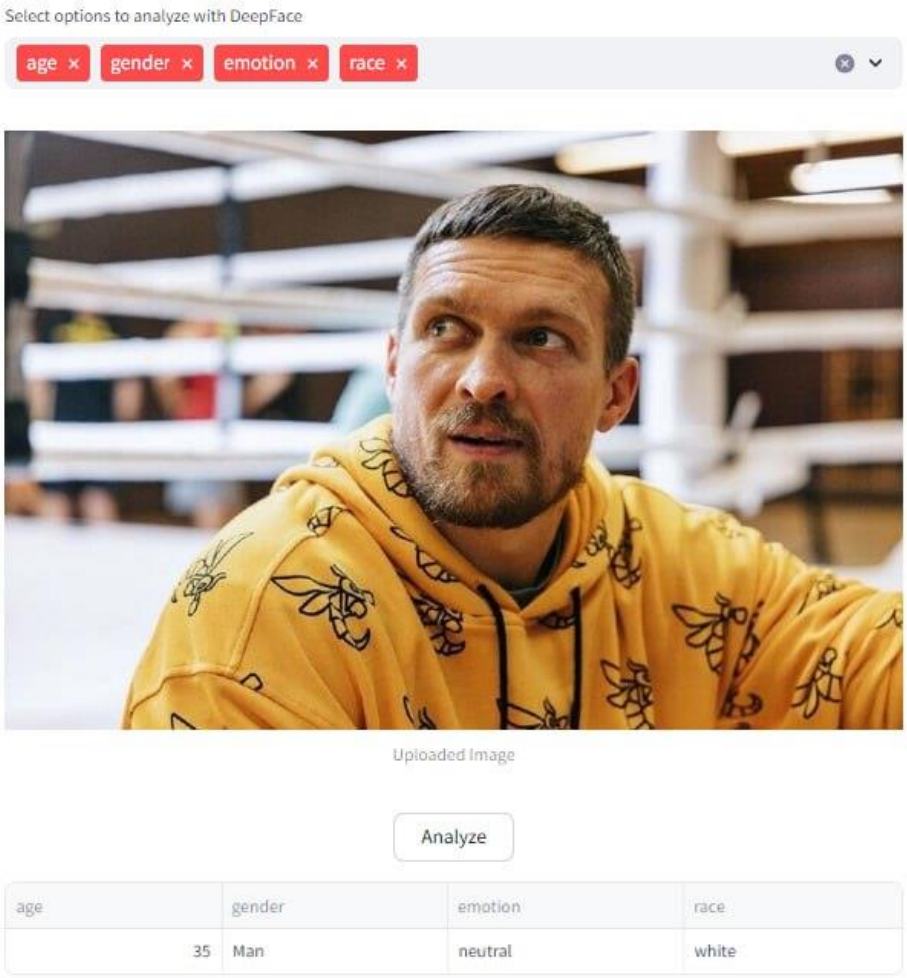

Рисунок 4.5 – Сторінка з розпізнаванням ознак (вік, стать, емоції та раса) на основі даних обличчя

На рис. 4.6 – 4.7 представлена наступна секція, де відбувається розпізнавання особи людини за зображенням. На рис. 4.6 відображений інтерфейс застосунка для розпізнавання обличчя. На фотографії видно, що обличчя було виявлено та обрамлено червоним прямокутником, який позначає область, де програмне забезпечення визначило обличчя. На рис. 4.7 показані результати аналізу. Відображене обличчя було вирізано та

збільшено для кращої візуалізації. Під зображенням обличчя розміщена таблиця з результатами розпізнавання.

Таблиця містить список потенційних збігів із базою даних, і для кожного збігу вказані ім'я, опис, дистанція (distance) та схожість (similarity).

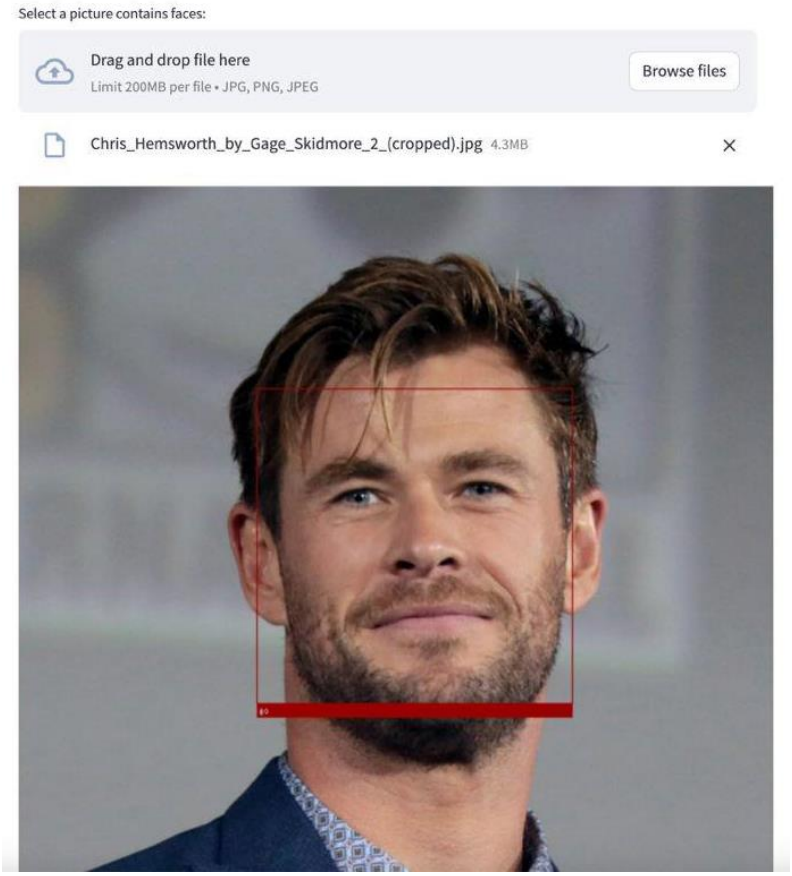

Рисунок 4.6 – Завантажене зображення для розпізнавання (id = 0)

Distance у цьому контексті означає косинусну відстань між вектором ознак обличчя, отриманим із завантаженого зображення, та вектором ознак з бази даних. Ця відстань вимірює схожість двох облич, де менша відстань означає більшу схожість між обличчями.

Іншими словами, чим менше значення distance, тим більше схожим є обличчя з бази даних на завантажене зображення. Кріс Хемсворт має найменше значення дистанції і найвищий рівень збігу, що вказує на високу ймовірність того, що це його обличчя.

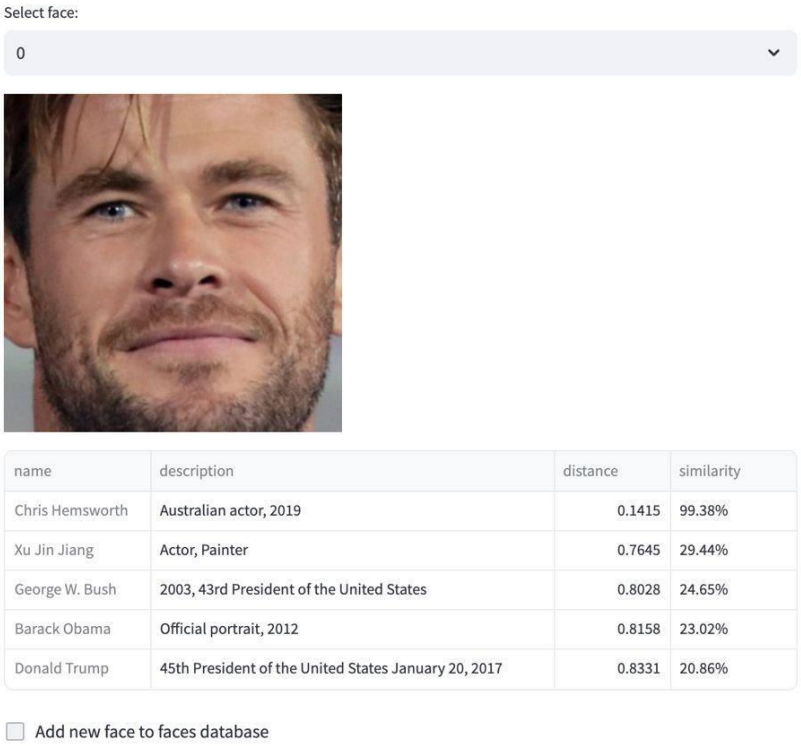

Рисунок 4.7 – Результати розпізнавання обличчя на зображенні

На рисунку 4.8 представлена секція з розпізнаванням обличчя з використанням потокового відео з веб-камери користувача системи.

Для сканування обличчя користувачів застосовується вбудована відеокамера, з якої відбувається верифікація клієнта.

Для відображення верифікації по фото, додана база даних Pandas для вже доданих облич або для тих, що будуть додані в процесі роботи системи.

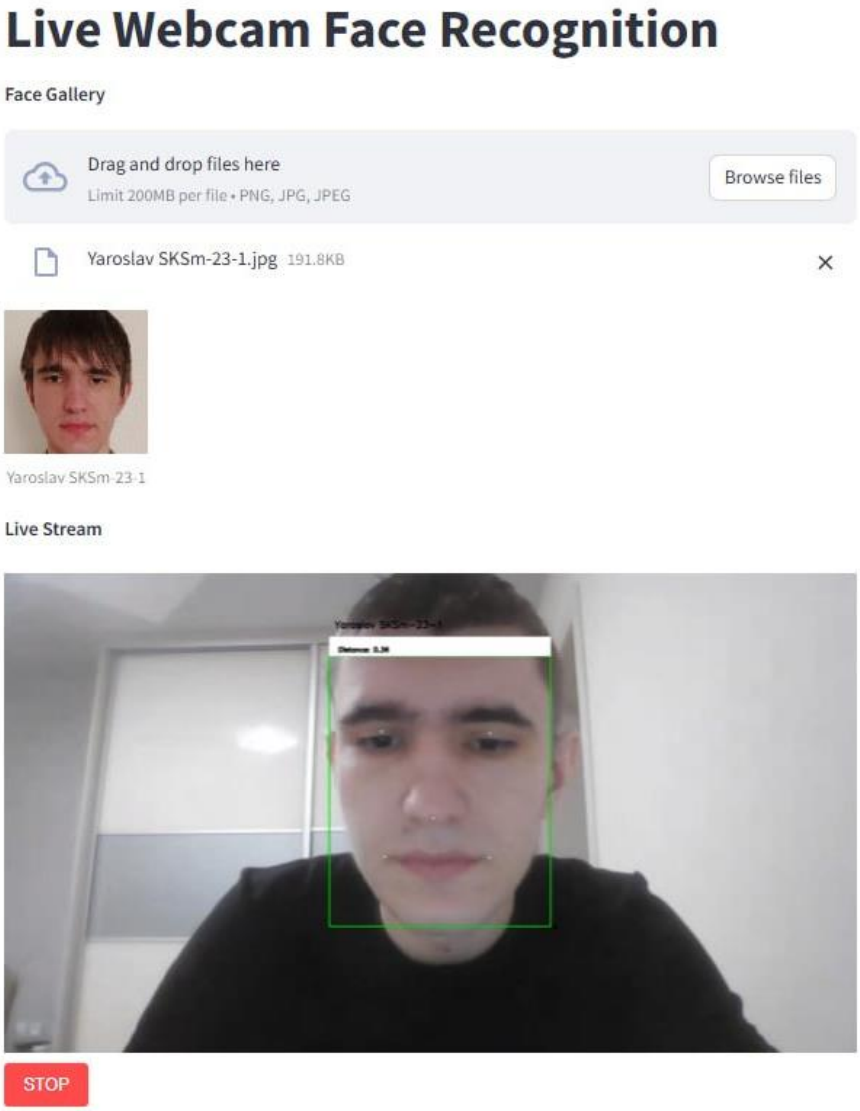

Рисунок 4.8 – Результати розпізнавання з використанням веб-камери

ВИСНОВКИ

Кваліфікаційна робота присвячена створенню системи біометричного розпізнавання обличчя з використанням технологій машинного навчання, яка продемонструвала ефективність використання згорткових нейронних мереж з метою розпізнавання біометричних даних облич особистостей.

В першому розділі проведено аналіз предметної області біометричного розпізнавання, а також розглянуто основні методи, що використовуються для розробки таких систем, а саме: метод, заснований на геометричних характеристиках обличчя, метод головних компонентів та метод глибокого навчання. Також, визначено переваги і недоліки зазначених методів, розглянуто приклади існуючих систем біометричного розпізнавання облич.

В другому розілі проведено аналіз сучасних алгоритмічних рішень, які використовуються для розробки систем біометричного розпізнавання. В ході аналізу було встановлено, що правильне налаштування згорткової нейронної мережі, включно з правильним вибором структури та параметрів навчання, критично впливає на продуктивність системи. Також було виявлено, що належна підготовка даних, включаючи попередню обробку зображень і аугментацію, покращує здатність мережі до узагальнення на основі нових даних.

В третьому розділі проведено огляд технологій, які були використані для розробки програмної системи. В розділі представлено блок-схему алгоритму розпізнавання обличчя, опис розроблених модулів системи, а також деталізовано логіку реалізації. Програмна реалізація системи розпізнавання розрблена за допомогою мови програмування Python з використанням фреймворку Streamlit, що являє собою веб-застосунок з платформою машинного навчання TensorFlow, середовищем розробки є PyCharm.

Для сканування обличчя користувачів застосовується вбудована відеокамера, з якої відбувається верифікація клієнта. Для відображення верифікації по фото, додана база даних Pandas для вже доданих облич або для тих, що будуть додані в процесі роботи системи.

В четвертому розділі представлено опис проведеного тестування основних модулів програмної системи на відповідність функціональним вимогам. Результати тестування продемонстрували правильність так коректність виконання поставленого завдання.

В результаті проведеного дослідження та розробки системи можна стверджувати, що розширення можливостей та оптимізація систем біометричного розпізнавання облич з використанням згорткових нейронних мереж є перспективним напрямком та дає можливості для подальших досліджень і використання у сфері безпеки даних та персоналізації сервісів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Facial Recognition: how it works and its safety. [Електронний ресурс]. – Режим доступу: https://www.signicat.com/blog/face-recognition (дата звернення: 01.06.2024).

2. Мілька Я.Ю. Система розпізнавання біометричних даних обличчя з використанням технологій машинного навчання. / Я.Ю Мілька // 28-й Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті». Зб. матеріалів форуму. Т. 6. – Харків: ХНУРЕ. 2024. – С. 290-292.

3. Face Recognition Software. [Електронний ресурс]. – Режим доступу: https://www.aware.com/facial-recognition/ (дата звернення: 01.06.2024).

4. Yudita S. I., Mantoro T., Ayu M. A. Deep Face Recognition for Imperfect Human Face Images on Social Media using the CNN Method. / S. I. Yudita, T. Mantoro and M.A. Ayu // 2021 International Conference of Computer Engineering, Depok, Indonesia. – 2021. – P. 412-417. – Режим доступу: https://ieeexplore.ieee.org/document/9649317 (дата звернення: 01.06.2024).

5. Mantoro T., Ayu M. A. Multi-Faces Recognition Process Using Haar Cascades and Eigenface Methods. / T. Mantoro and M.A. Ayu // 2018 International Conference on Multimedia Computing and Systems, Rabat, Morocco. – 2018. – P. 1-5. [Електронний ресурс]. – Режим доступу: https://ieeexplore.ieee.org/document/8525935 (дата звернення: 01.06.2024).

6. Sikarwar A., Chandra H., Ram I. Real-Time Biometric Verification and Management System Using Face Embeddings. / A. Sikarwar, H. Chandra and I. Ram. // IEEE 17th India Council International Conference (INDICON), New Delhi, India. – 2020. – P. 1-4, [Електронний ресурс]. – Режим доступу: https://ieeexplore.ieee.org/document/9342551 (дата звернення: 01.06.2024).

7.. An approach for Face Detection and Face Recognition using OpenCV and Face Recognition Libraries in Python. / K. Sravani, K. S. Priya and V. Bhavani

// International Conference Communication Systems., Coimbatore, India. – 2023. – P. 1274-1278. – Режим доступу: https://ieeexplore.ieee.org/document/10113066 (дата звернення: 01.06.2024).

8. The development and comparison of face recognition algorithms based on different technical characteristics. / Z. Guo // International Conference on Computer Vision, Image and Deep Learning., Chongqing, China – 2020. – P. 6-10. – Режим доступу: https://ieeexplore.ieee.org/document/9270541 (дата звернення: 01.06.2024).

9. Meng X., Yan Y., Chen S., Wang H. A Cascaded Noise-Robust Deep CNN for Face Recognition. / X. Meng, Y. Yan, S. Chen and H. Wang // IEEE International Conference on Image Processing., Taipei, Taiwan. – 2019. – P. 3487-3491. – Режим доступу: https://ieeexplore.ieee.org/document/8803443 (дата звернення: 01.06.2024).

10. Lin H., Ma H., Gong W., Wang C. Non-frontal face recognition method with a side-face-correction generative adversarial networks. / H. Lin, H. Ma, W. Gong and C. Wang // International Conference on Computer Vision, Image and Deep Learning., Changchun, China. – 2022. – P. 563-567. – Режим доступу: https://ieeexplore.ieee.org/document/9825237 (дата звернення: 01.06.2024).

11. Rajput S. S., Arya K. V. CNN Classifier based Low-resolution Face Recognition Algorithm. / S. S. Rajput and K. V. Arya // International Conference on Emerging Frontiers in Electrical and Electronic Technologies., Patna, India. – 2020. – P. 1-4. - Режим доступу: https://ieeexplore.ieee.org/document/9187001 (дата звернення: 01.06.2024).

12. Winarno E., Husni Al Amin I., Februariyanti H., Adi P. W., Hadikurniawati W., Anwar M. T. Attendance System Based on Face Recognition System Using CNN-PCA Method. / E. Winarno, I. Husni Al Amin, H. Februariyanti, P. W. Adi, W. Hadikurniawati and M. T. Anwar // International Seminar on Research of Information Technology and Intelligent Systems., Yogyakarta, Indonesia. – 2019. – P. 301-304. – Режим доступу: https://ieeexplore.ieee.org/document/9034596 (дата звернення: 01.06.2024).

13. Dwijayanti S., Abdillah R. R., Hikmarika H., Husin Z., Suprapto B. Y. Facial Expression Recognition and Face Recognition Using a Convolutional Neural Network. / S. Dwijayanti, R. R. Abdillah, H. Hikmarika, Z. Husin and B. Y. Suprapto // International Seminar on Research of Information Technology and Intelligent Systems. Yogyakarta, Indonesia. – 2020. – P. 621-626. – Режим доступу: https://ieeexplore.ieee.org/document/9315513 (дата звернення: 01.06.2024).

14. Nandre J., Rai S.,. Kanawade B. R. Comparative Analysis of Transfer Learning CNN. / J. Nandre, S. Rai and B. R. Kanawade // International Conference on Technologies. Hubli, India. – 2022. – P. 1-6. – Режим доступу: https://ieeexplore.ieee.org/document/9847946 (дата звернення: 01.06.2024).

15. He X., Ding F. Efficient Face Recognition Method Based on CNN. / X. He and F. Ding // International Conference on Computer Applications. Shenyang, China. – 2023. – P. 1788-1791. – Режим доступу: https://ieeexplore.ieee.org/document/10076242 (дата звернення: 01.06.2024).

16. Miakshyn O., Anufriiev P., Bashkov Y. Face Recognition Technology Improving Using Convolutional Neural Networks. / O. Miakshyn, P. Anufriiev and Y. Bashkov // International Conference on Advanced Trends in Information Theory., Kyiv, Ukraine. – 2021. – P. 116-120. – Режим доступу: https://ieeexplore.ieee.org/document/9678722 (дата звернення: 01.06.2024).