Харківський національний університет імені В.Н. Каразіна

Факультет комп'ютерних наук

Безпека інформаційних систем і технологій

«Допущено до захисту»

Зав.кафедрою БІСТ

Сватовський І.І._____
« » червня 2023р.

Пояснювальна записка до

кваліфікаційної роботи бакалавра

спеціальність: 125 Кібербезпека

на тему: «Розробка методу біометричної аутентифікації на основі розпізнавання облич»

оцінка « »

Голова ЕК

Лемешко О.В._____

Керівник проф. Кузнецов О.О.
(прізвище та ініціали/підпис)

Рецензент проф. Краснобаєв В.А.
(прізвище та ініціали/підпис)

Виконавець студентка групи КБ-41

Халамай С.С.
(прізвище та ініціали/підпис)

Харків – 2023

РЕФЕРАТ

Пояснювальна записка містить 62 сторінки, 23 рисунки, 3 таблиці, 1 додаток, 25 джерел.

Метою дипломної роботи є дослідження та розробка методу біометричної аутентифікації на основі розпізнавання облич.

Об'єктом дослідження дипломної роботи є процес біометричної аутентифікації, зокрема розпізнавання обличчя, що включає в себе методи, алгоритми та технології для виявлення, вилучення ознак, моделювання та порівняння обличчя людини з метою її ідентифікації або аутентифікації.

Предметом розробки є метод біометричної аутентифікації, який базується на розпізнаванні облич. Цей метод повинен правильно розпізнавати живі обличчя і виявляти спроби використання цифрових зображень або відео для обхідного доступу до системи.

Методи дослідження: аналіз предметної області та постановка задачі, дослідження аналогічних рішень сучасних систем розпізнавання обличчя, розробка та тестування Liveness Detection.

Результатами проведеної роботи є метод біометричної аутентифікації на основі розпізнавання облич, оцінка його ефективності та можливості його застосування в реальних умовах. Також аналіз про точність розпізнавання облич від отриманих даних.

Ключові слова: РОЗПІЗНАВАННЯ ОБЛИЧЧЯ, БІОМЕТРИЧНА АУТЕНТИФІКАЦІЯ, МЕТОДИ БІОМЕТРИЧНОЇ АУТЕНТИФІКАЦІЇ, МАШИННЕ НАВЧАННЯ, ШТУЧНИЙ ІНТЕЛЕКТ, ГЛИБИННЕ НАВЧАННЯ.

ABSTRACT

The explanatory note contains 62 pages, 23 figures, 3 tables, 1 annex, 25 sources.

The aim of the thesis is to research and develop a method of biometric authentication based on facial recognition.

The subject matter is the process of biometric authentication, specifically face recognition. It encompasses methods, algorithms, and technologies for detecting, extracting features, modeling, and comparing a person's face for the purpose of identification or authentication.

The scope of the study is a method of biometric authentication that is based on facial recognition. This method should accurately recognize live faces and detect attempts to use digital images or videos for bypassing system access.

Research methods: analysis of the subject area and problem statement, research of similar solutions in modern face recognition systems, development and testing of Liveness Detection.

The results of the conducted work include the development of a biometric authentication method based on facial recognition, the evaluation of its effectiveness, and the possibilities of its application in real-world conditions. In addition, the analysis of face recognition accuracy from the obtained data.

Keywords: FACE RECOGNOTION, BIOMETRIC AUTHENTIFACATION, BIOMETRIC AUTHENTICATION METHODS, MACHINE LEARNING, ARTIFICIAL INTELLIGENCE, DEEP LEARNING.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

CV	-	комп’ютерний зір (Computer Vision)
API	-	прикладний програмний інтерфейс (application programming interface)
FRS	-	система розпізнавання обличчя (Facial Recognition System)
2D	-	Двовимірний (Two-Dimensional)
3D	-	Тривимірний (Three-Dimensional)
LBP	-	локальний бінарний шаблон (Local Binary Pattern)
SVM	-	метод опорних векторів (Support Vector Machines)
GPU	-	графічний процесор (Graphics Processing Unit)
ROI	-	області інтересу (Regions of Interest)
CNN	-	згорткова нейронна мережа (Convolutional Neural Network)
CONV	-	згортковий шар (Convolutional Layer)
RELU	-	функція активації (Rectified Linear Unit)
POOL	-	пулінговий шар (Pooling Layer)
FC	-	повнозв'язний шар (Fully Connected Layer)
AI	-	штучний інтелект (Artificial Intelligence)
ML	-	машинне навчання (Machine Learning)
DL	-	глибоке навчання (Deep Learning)
TP	-	вірно позитивний результат (True Positive)
FN	-	помилковий негативний (False Negative)
FP	-	помилковий позитивний (False Positive)
TN	-	вірно негативний (True Negative)
DCNN	-	згорткова нейронна мережа з декількома шарами (Deep Convolutional Neural Network)

ВСТУП

У сучасному цифровому світі забезпечення надійності ідентифікації користувачів стає все більш актуальним завданням. Захист особистої інформації, фінансових ресурсів та цифрових активів вимагає інноваційних підходів, які б забезпечили високий рівень безпеки і зручності для користувачів. У цьому контексті біометричні технології набувають все більшої популярності та значущості.

Одним із найпоширеніших біометричних ознак, які можуть бути використані для ідентифікації особи, є розпізнавання облич. Людське обличчя має унікальні особливості, такі як форма обличчя, положення очей, розташування вуст, та інші анатомічні деталі, які варіюються від особи до особи. Ці унікальні характеристики можуть бути використані для створення ефективної системи біометричної аутентифікації. Завдяки швидкому розвитку комп'ютерного зору та машинного навчання, розпізнавання облич стало можливим з вражаючою точністю та швидкістю.

Розробка методу біометричної аутентифікації на основі розпізнавання облич має потенціал зробити наші системи безпеки більш надійними та зручними для користувачів.

Використання обличчя як біометричного індикатора може знизити ризик шахрайства та несанкціонованого доступу до особистих даних. Крім того, цей метод може бути використаний в різних сферах, включаючи фізичний доступ до приміщень, електронну комерцію, банківські операції, мобільні пристрої та багато інших.

Метою даного дослідження є розробка методу біометричної аутентифікації на основі розпізнавання облич, який поєднає високу точність і швидкість ідентифікації. Також важливим є створення системи, яка здатна ефективно визначати особу на основі зображення обличчя, використовуючи штучний інтелект. При цьому метод повинен коректно розпізнавати живе

обличчя та виявляти спроби використання цифрових зображень або відео для обхідного доступу до системи.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

У цьому розділі буде проводитись аналіз предметної області, пов'язаної з біометричною аутентифікацією на основі розпізнавання облич, буде розглянута різниця між аутентифікацією та верифікацією; існуючі підходи, методи та технології, які використовуються в даній області та буде поставлена задача дослідження.

1.1 Основні поняття та визначення

Біометрія дозволяє ідентифікувати та аутентифікувати людину на основі впізнаваних, перевірених, унікальних та конкретних даних.

Біометрична аутентифікація та ідентифікація є двома поняттями, пов'язаними з використанням біологічних характеристик для ідентифікації особи. Однак вони використовуються в різних контекстах та мають певні різниці.

Біометрична ідентифікація використовується для встановлення особи на основі її біологічних характеристик. В цьому процесі збираються біометричні дані – розпізнавання обличчя, відбитки пальців, сканування раковини вуха, розпізнавання шаблону сітківки або сканування пальцевого відбитка [1]. Ці дані порівнюються з збереженими в базі даних, щоб знайти відповідну особу або встановити її ідентичність. Біометрична ідентифікація використовується в багатьох сферах, таких як паспортні контрольні пункти, в'їзд на робоче місце або доступ до комп'ютерних систем.

З іншого боку, біометрична аутентифікація використовується для перевірки ідентичності особи на основі її біологічних характеристик. В цьому процесі користувач представляє свої біометричні дані для порівняння зі збереженими в базі даних. Якщо біометричні дані користувача збігаються зі збереженими даними, то аутентифікація вважається успішною, і користувач отримує доступ до системи або послуги. Біометрична аутентифікація

використовується, наприклад, для розблокування смартфонів за допомогою відбитків пальців або розпізнавання обличчя.

Процес біометричної аутентифікації зазвичай включає наступні кроки:

1) реєстрація – під час цього кроку користувач надає свої біометричні дані, наприклад, сканує свій відбиток пальця, створює шаблон обличчя або записує голосовий зразок. Ці дані потім зберігаються у базі даних для подальшої порівняння;
2) виявлення – коли користувач намагається отримати доступ до системи або пройти ідентифікацію, його біометричні дані зчитуються за допомогою спеціального пристрою, такого як сканер відбитків пальців або камера для розпізнавання обличчя;
3) порівняння – зчитані біометричні дані порівнюються зі збереженими даними в базі даних, якщо отримані дані відповідають даним, зареєстрованим в системі, то аутентифікація вважається успішною;
4) підтвердження – у разі успішної аутентифікації користувачу надається доступ до системи або йому підтверджується його особистість [2].

Біометрична аутентифікація дозволяє забезпечити високий рівень безпеки та зручності для користувачів, оскільки біометричні дані важко підробити або втратити [1]. Однак, варто враховувати проблеми, пов'язані з приватністю та збереженням біометричних даних, тому важливо використовувати надійні методи захисту цих даних.

Різниця між біометричною ідентифікацією та аутентифікацією полягає в тому, що ідентифікація використовується для знаходження відповідної особи на основі біометричних даних, тоді як аутентифікація використовується для перевірки ідентичності особи на основі біометричних даних, які вона надає.

1.2 Типи та методи біометричної аутентифікації

Біометрична аутентифікація використовує фізичні або поведінкові характеристики людини для ідентифікації та перевірки її особистості.

Існують два основних типи біометрії: фізіологічна біометрія та поведінкова біометрія (див. рисунок 1.1) [1]. Фізіологічна біометрія залежить

від унікальних фізіологічних ознак людини для розпізнавання або підтвердження особи, тоді як поведінкова біометрія використовує унікальні особливості поведінки людини.

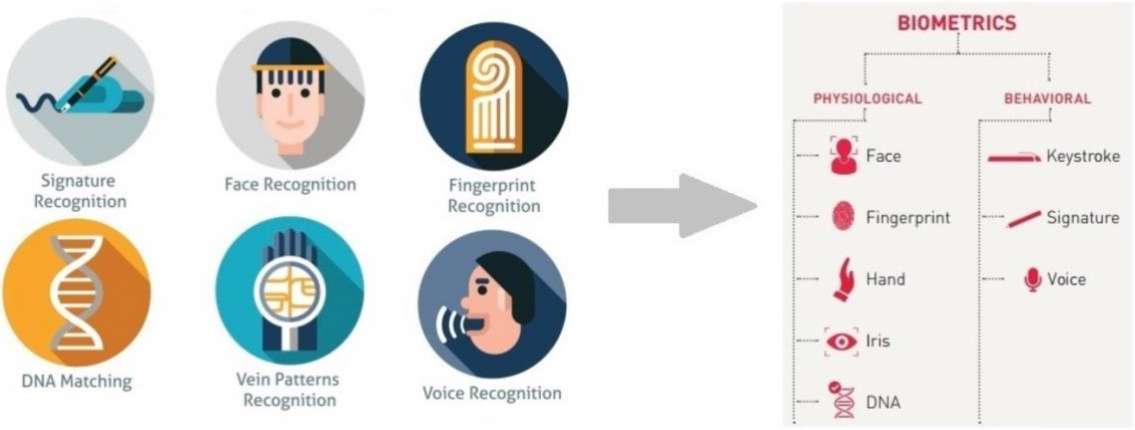

Рисунок 1.1 – Види біометрії і приклади фізіологічних і поведінкових вимірювань

Біометрична аутентифікація стає все більш популярною технологією, оскільки забезпечує вищий рівень надійності порівняно з традиційними методами аутентифікації, такими як паролі або PIN-коди. Нижче наведено перелік існуючих методів біометричної аутентифікації:

- відбитки пальців – використовуються унікальні особливості відбитків пальців для ідентифікації особи, датчики відбитків пальців захоплюють образ пальця, алгоритми порівнюють його зі збереженими в базі даних;
- розпізнавання обличчя – захоплення і аналіз геометричних особливостей обличчя людини, таких як форма обличчя, розташування очей, ніздрі та рота, цей метод використовується в системах розблокування смартфонів, контролю доступу та відеоспостереження;
- сканування радужної оболонки ока – ідентифікація особи за унікальними особливостями радужки ока, використовується високоточне сканування з використанням інфрачервоного світла;
- голосова аутентифікація – визначення особливостей голосу, таких як тембр, тональність, ритм та інтонація, голос можна використовувати як унікальну біометричну ознаку для ідентифікації особи;

- розпізнавання почерку – аналіз особливостей почерку та рукопису, користувачі набирають текст або пишуть на планшеті, алгоритми аналізують особливості рукопису для ідентифікації особи;
- сканування структури вен – використання особливостей структури вен, зокрема на долонях або очах. Венозне розпізнавання вважається дуже надійним, оскільки вени є унікальними для кожної людини [1].

Ці методи можуть використовуватись окремо або в комбінації між собою для створення більш надійних систем біометричної аутентифікації. Крім того, важливо зберігати біометричні дані в безпечних системах, оскільки їх скомпрометування може мати серйозні наслідки для особистої безпеки користувачів.

1.3 Системи комп’ютерного зору

Комп'ютерний зір, також відомий як комп'ютерне зорове сприйняття (англ. CV, computer vision), є технологією, що розвиває комп'ютерні системи для виявлення, стеження та класифікації об'єктів.

Застосування комп'ютерного зору є досить широкими і розповсюдженими в різних галузях. Системи комп'ютерного зору використовуються для автоматичного розпізнавання образів і класифікації об'єктів на зображеннях або відео. Це може застосовуватися у сферах медицини (розпізнавання пухлин на зображеннях рентгенівських), безпеки (виявлення об'єктів на контрольних пунктах аеропорту) та багатьох інших.

Використання штучного інтелекту та машинного навчання значно підняло аналіз, прогнозування та розпізнавання на новий рівень. Одним з провідних напрямків є розпізнавання облич, і в цій області було досягнуто великих успіхів.

Наприклад, технологія Face ID, яка була представлена на заході Apple Special Event у вересні 2017 року, є відмінним прикладом використання розпізнавання облич [4]. Вона зараз активно використовується у соціальних мережах для автоматичного позначення людей на фотографіях. Facebook є

однією з компаній-лидерів у цій сфері, і їх алгоритм показує ефективність розпізнавання на рівні 93%.

Результатом широкого використання технології Face ID стала поява різних бібліотек та API для розпізнавання облич. Існують спеціалізовані рішення, призначені для певних сфер або універсальні, написані на різних мовах програмування або з підтримкою різних мов. Часто вибір серед цього різноманіття варіантів може бути важким завданням, щоб знайти найкраще підходящий для конкретної проблеми інструмент.

1.4 Постановка задачі дослідження

Ця робота присвячена проблематиці комп'ютерного зору, зокрема розпізнаванню облич на основі бібліотеки OpenCV [5]. В ході дослідження розглядатиметься доцільність використання при створені систем розпізнавання та будуть оцінюватися переваги і недоліки цієї бібліотеки.

Задача дослідження полягає у розробці методу біометричної аутентифікації, який базується на технології розпізнавання облич. Основна мета полягає в розробці методу, який забезпечує автоматичну ідентифікацію особи на основі аналізу її обличчя та аутентифікацію, здатну відповідати на виклики, пов'язані з захистом від підробки. При цьому метод повинен коректно розпізнавати живе обличчя та виявляти спроби використання цифрових зображень або відео для обхідного доступу до системи.

2 ОСНОВНІ АСПЕКТИ ТЕХНОЛОГІЇ РОЗПІЗНАВАННЯ ОБЛИЧЧЯ

У даному розділі будуть розглянуті основні принципи роботи систем розпізнавання обличчя, детальна характеристика технологій розпізнавання обличчя, аналіз відомих рішень щодо цих технологій, а також актуальность їх використання. Також будуть проведені дослідження ринку систем розпізнавання обличчя для отримання більш повної карти сучасного стану цієї галузі.

2.1 Принципи роботи систем розпізнавання облич

Системи розпізнавання облич (англ. FRS, facial recognition system) використовуються для автоматичного визначення та ідентифікації людських облич на зображеннях або відео [6]. Нижче наведена таблиця 2.1, що описує основні принципи роботи таких систем.

Таблиця 2.1 – Основні принципи роботи FRS

Принцип	Опис
Збір даних	Система отримує вхідне зображення облич людей, яке може бути отримане з різних джерел, наприклад, відеокамер, фотографій або відеопотоків.
Попередня обробка	Вхідне зображення може бути попередньо оброблене для покращення якості зображення, включаючи розмиття, фільтрацію шуму або зміну розміру. Це допомагає покращити точність подальшого розпізнавання.
Виділення облич	Система застосовує алгоритми комп'ютерного зору для виділення облич на зображенні. Це може включати використання методів виявлення облич, які базуються на розпізнаванні контурів, знаходженні областей зі шкірою або використанні нейронних мереж.

Продовження таблиці 2.1 – Основні принципи роботи FRS

Ознаки та вектори облич	Після виділення облич система аналізує їх особливості та ознаки, такі як форма обличчя, розміщення очей, носа, рота тощо. Ці ознаки можуть бути використані для побудови векторів облич, що представляють кожне обличчя у просторі ознак.
Класифікація	Отримані вектори облич можуть бути подані на вхід до алгоритмів машинного навчання або нейронних мереж для класифікації. Система може порівнювати вектори облич з відомими шаблонами, щоб ідентифікувати особу або виконувати пошук у базі даних.
Відстеження	У деяких системах розпізнавання облич може бути реалізовано відстеження облич протягом часу. Це дозволяє відслідковувати рухи та зміни положення облич на відео чи в потоці зображень.
Ідентифікація та аутентифікація	За допомогою порівняння векторів облич із відомими шаблонами, система може здійснювати верифікацію, тобто перевірку, чи належить обличчя певній особі, або ідентифікацію, тобто встановлення особи на зображенні.

Важливо відзначити, що робота систем розпізнавання облич може варіюватися в залежності від точності алгоритмів, доступної інформації про особи у базі даних, якість зображень, умов освітлення та інших факторів. Деякі системи також можуть використовувати додаткові методи, наприклад, аналіз руху або додаткові датчики для підвищення точності розпізнавання облич [7].

2.2 Характеристика 2D, 3D та технологій аналітики обличчя

Розпізнавання обличчя за допомогою 2D і 3D та технологій аналітики обличчя є широко використовуваним методом в багатьох сферах, включаючи

безпеку, контроль доступу, відеоспостереження, розпізнавання особистості та багато іншого.

2D розпізнавання обличчя базується на аналізі двовимірних зображень обличчя. За допомогою алгоритмів комп'ютерного зору, таких як методи границь, методи кращих признаків або нейронні мережі, 2D системи розпізнавання обличчя можуть ідентифікувати та розрізняти особи на зображеннях [8]. Ці системи широко використовуються для розпізнавання обличчя в фотографіях, відеозаписах та навіть в реальному часі з використанням веб-камер.

3D розпізнавання обличчя використовує технології, які можуть аналізувати тривимірну форму обличчя, зазвичай за допомогою 3D-сканування або використанням спеціальних датчиків, таких як датчики структури світла або стереокамер [8]. Ці системи дозволяють отримувати детальну інформацію про форму обличчя та його геометрію, що може бути корисною для більш точного розпізнавання та аутентифікації особи.

Аналітика обличчя включає в себе додаткові функції, такі як визначення емоцій, вікова група, стать, розпізнавання виразів обличчя та інші характеристики. Вона базується на алгоритмах, які аналізують різні аспекти обличчя, такі як форма очей, рота, бровей, показники кольору шкіри та інші ознаки, для встановлення цих характеристик [9].

2.3 Аналіз аналогічних рішень

Аналізуючи відомі технології розпізнавання обличчя можна відзначити наступні:

- Apple Face ID;
- Facebook DeepFace;
- Microsoft Face API.

Загалом, всі три технології мають вражаючі можливості розпізнавання обличчя. Facebook DeepFace визначається високою точністю розпізнавання, Apple Face ID – безпекою та використанням тривимірної моделі обличчя, а

Microsoft Face API – широким спектром сервісів для аналізу обличчя та розпізнавання емоцій.

2.3.1 Apple Face ID

Face ID від Apple є технологією розпізнавання обличчя, використовуваною для ідентифікації користувачів на пристроях Apple, зокрема на iPhone та iPad [4]. Особливості Face ID:

1) Face ID використовує технологію TrueDepth, яка включає передню камеру з інфрачервоним проектором та датчиком глибини;
2) вона створює точну тривимірну модель обличчя користувача і використовує її для порівняння зі збереженою моделлю;
3) Face ID відомий своєю високою безпекою, оскільки використовується унікальна тривимірна модель обличчя, яка складна для підробки.

Рисунок 2.1 – Принцип роботи системи Apple Face ID

2.3.2 Facebook DeepFace

Facebook DeepFace є одним з провідних алгоритмів розпізнавання обличчя. Він використовує нейронну мережу глибинного навчання для аналізу та порівняння обличчя на зображеннях [10]. Основні особливості DeepFace:

- модель DeepFace має близько 120 мільйонів параметрів і навчається на великій кількості обличчєвих зображень з Facebook [10];
- вона використовує конволюційні шари для виявлення особливостей обличчя та глибокі повнозв'язні шари для подальшого аналізу;
- DeepFace має вражаючу точність розпізнавання, здатну впоратися з складними завданнями, такими як визначення того, чи належить два обличчя до однієї особи.

2.3.3 Microsoft Face API

Microsoft Face API – це набір сервісів, наданих Microsoft для розпізнавання обличчя. Він надає розширені можливості аналізу обличчя та розпізнавання емоцій [11]. Основні особливості Microsoft Face API:

- Microsoft Face API забезпечує розпізнавання обличчя в реальному часі та можливість визначення ключових точок на обличчі, таких як очі, ніс і рот;
- він також надає функції визначення віку, статі, емоцій та інших характеристик на основі обличчя [11];
- Microsoft Face API використовується для широкого спектру застосувань, включаючи системи безпеки, аналіз настрою користувачів, автоматичне тегування фотографій тощо.

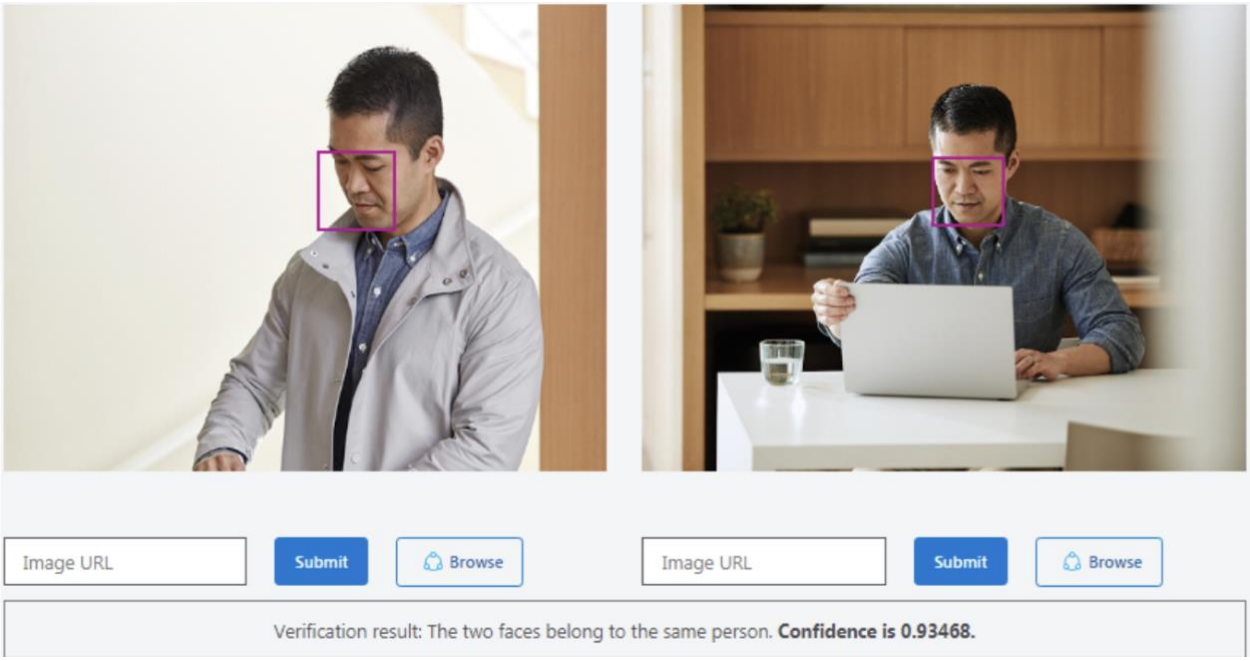

Рисунок 2.2 – Результат розпізнавання за допомогою Microsoft Face API

2.4 Обмеження традиційного підходу розпізнавання

Традиційні методи розпізнавання обличчя використовують такі підходи, як використання власних обличчя для формування базового набору зображень та низькорозмірне представлення зображень як зображено на рисунку 2.3, за допомогою алгебраїчних обчислень [12]. Потім розробники алгоритмів вирушили різними шляхами. Деякі з них акцентували увагу на відмінних рисах

облич та їх просторовому розташуванні відносно один одного. Деякі експерти також досліджували способи розбиття зображень для їх порівняння з шаблонами.

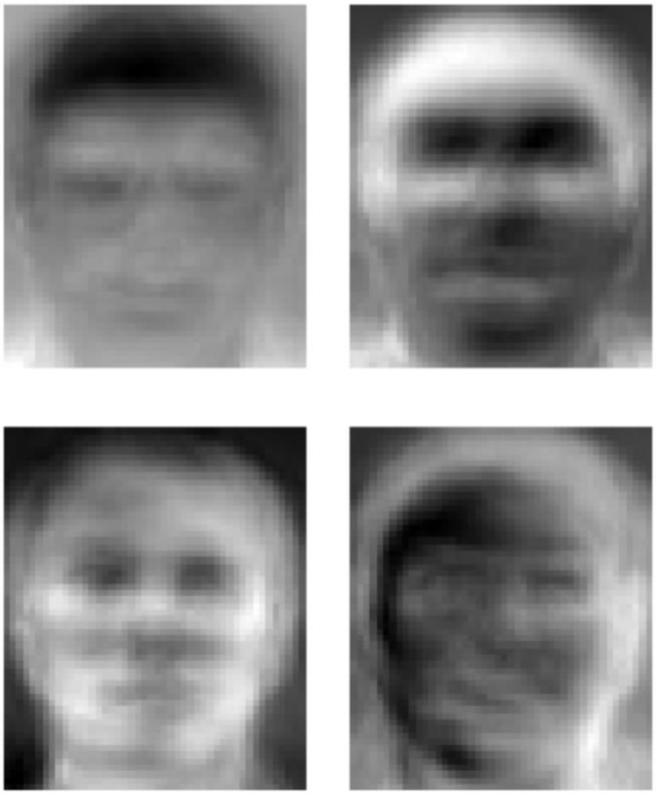

Рисунок 2.3 – Перші спроби реалізації розпізнавання обличчя

Зазвичай автоматизований алгоритм розпізнавання обличчя намагається відтворити спосіб, яким людина розпізнає обличчя. Однак, людські можливості дозволяють нам зберігати всю необхідну візуальну інформацію у мозку та використовувати її за потреби. У випадку комп'ютера все набагато складніше [12]. Щоб ідентифікувати людське обличчя, автоматизовані системи повинні мати доступ до досить повної бази даних та запитувати її для порівняння з тим, що вони бачать.

Традиційний підхід дозволив розробити програмне забезпечення для розпізнавання облич, яке виявило себе задовільно у багатьох випадках. Автоматизоване розпізнавання облич здобуло популярність завдяки безконтактному процесу ідентифікації. Підтвердження особи таким чином

відбувається швидко і непомітно, і також спричиняє відносно менше скарг, опозиції та конфліктів.

Серед переваг, які варто відзначити, є швидкість обробки даних, сумісність та можливість імпорту даних з більшості відеосистем. У той же час, недоліки та обмеження традиційного підходу до розпізнавання облич також очевидні.

Спочатку слід зазначити, що технологія розпізнавання обличчя має низьку точність в умовах швидкого руху та поганого освітлення. Невдалим випадкам розпізнавання близнюків, а також прикладам, що виявили певні расові упередження, користувачі ставляться негативно. Слабким моментом було збереження конфіденційності даних. Час від часу недостатня гарантія приватності та дотримання громадянських прав ставала навіть причиною заборони використання таких систем. Виникла потреба як у підвищенні точності біометричних систем, так і у додаванні до них функції виявлення цифрових або фізичних атак [13].

Проте, традиційний підхід до розпізнавання обличчя в основному вичерпав свій потенціал. Він не дозволяє використовувати дуже великі набори даних облич, а також не забезпечує навчання та налаштування систем ідентифікації з прийнятною швидкістю [13].

2.5 Актуальність використання біометричного розпізнання обличчя

Сьогодні все більше галузей впроваджують біометричне розпізнавання обличчя в щоденну роботу. Процес розпізнавання обличчя використовується на різних етапах, починаючи від розблокування телефону та проходження через безпеку в аеропорту, і до можливості покупки товарів у магазинах.

Технологія розпізнання обличчя залишається актуальною у сучасному світі з кількох причин:

- безпека – розпізнання обличчя широко використовується для забезпечення безпеки урядових, комерційних та громадських приміщень. Вона може використовуватися для контролю доступу до обмежених зон, відстеження підозрілих осіб та виявлення злочинців.

Технологія розпізнання обличчя допомагає збільшити рівень безпеки та запобігти можливим загрозам;

- аутентифікація – розпізнання обличчя може бути використано для біометричної аутентифікації користувачів. Це означає, що люди можуть використовувати своє обличчя для розблокування пристроїв, авторизації у банківських системах, мобільних додатках та інших онлайн-платформах. Це зручний і безпечний спосіб ідентифікації особи, оскільки обличчя важко підробити або вкрасти;
- покращення користувацького досвіду – технологія розпізнання обличчя може використовуватися для покращення користувацького досвіду у різних сферах. Наприклад, вона може використовуватися для ідентифікації клієнтів у роздрібних магазинах, готелях або ресторанах для персоналізації обслуговування. Технологія також може бути застосована у сферах медицини, освіти та розваг для створення інтерактивних досвідів для користувачів;
- виявлення емоцій та настроїв – розпізнання обличчя може бути використано для виявлення емоцій та настроїв людини. Це може бути корисним у багатьох галузях, таких як маркетинг та реклама, дослідження ринку, психологія та психіатрія. Аналізуючи вирази обличчя, можна отримати цінну інформацію про реакції людей на певні події, продукти або послуги.

Необхідно зазначити, що використання технології розпізнання обличчя також викликає етичні питання, пов'язані з приватністю та можливими зловживаннями. Важливо забезпечувати адекватний рівень захисту персональних даних та використовувати цю технологію з врахуванням прав та свобод особи.

2.5.1 Дослідження ринку Facial Recognition

Global Market Insights – це незалежна дослідницька компанія, яка спеціалізується на аналізі ринків і проведенні стратегічних досліджень. Вона надає детальні звіти, прогнози і аналітику про різні галузі та ринки, включаючи

технологічні інновації, ринки продуктів і послуг, фінансові ринки та інші. За їх даними, що вказані на рисунку 2.4, розмір ринку розпізнавання обличчя перевищив 3 мільярда доларів у 2019 році і прогнозується зростання на рівні понад 18% між 2020 і 2026 роками. Це стається через швидке поширення передових систем розпізнавання обличчя для забезпечення безпеки та ідентифікації [14].

Рисунок 2.4 – Звіт Global Market Insights дослідження ринку систем розпізнавання обличчя

Аналітики компанії Allied Market Research запевняють, що розмір глобального ринку розпізнавання обличчя становив 3,83 мільярда доларів у 2020 році і прогнозується досягти 16,74 мільярда доларів до 2030 року, зростаючи з річною зростанням на 16,0% від 2021 до 2030 року. На рисунку 2.5 вказано прогноз розвитку індустрії FRS за технологіями до 2030 року [15]. Більше того, точність систем розпізнавання обличчя значно поліпшилася протягом останнього десятиліття. Наприклад, за тестами, проведеними Національним інститутом стандартів та технологій у квітні 2020 року, найкращий алгоритм ідентифікації обличчя мав помилкову ставку всього 0,08%, що є великим покращенням по відношенню до 2014 року, коли найкращий алгоритм мав помилкову ставку 4,1% [15].

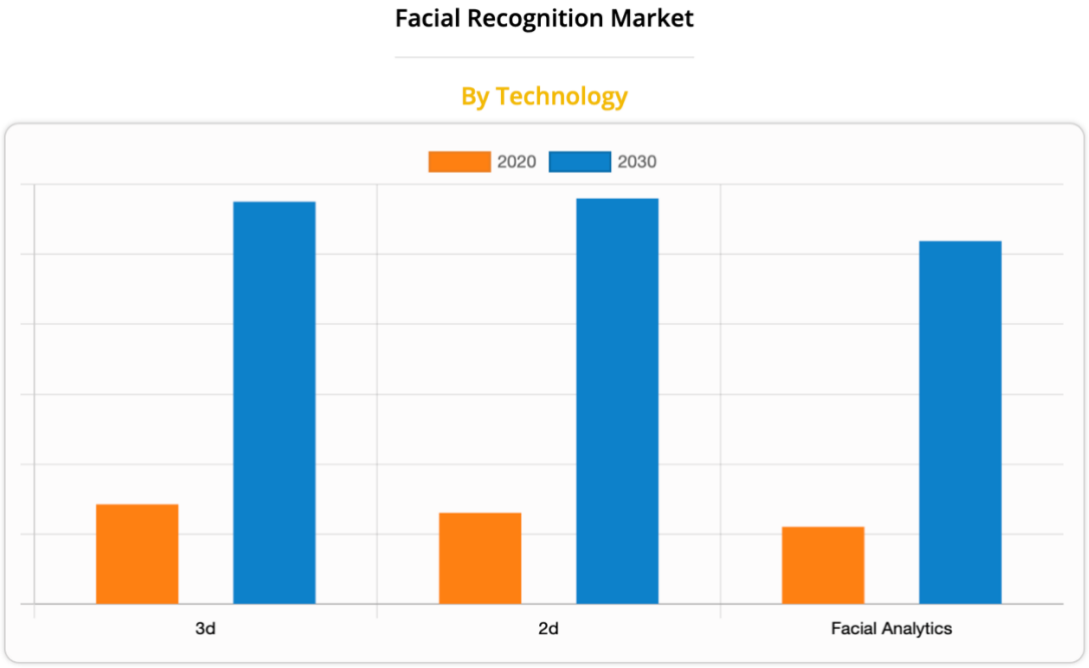

Рисунок 2.5 – Прогноз розподілу індустрії розпізнавання обличчя за технологіями на 2030 рік від компанії Allied Market Research

3 ЗАСТОСУВАННЯ ШТУЧНОГО ІНТЕЛЕКТУ ДЛЯ РОЗПІЗНАВАННЯ ТА ІДЕНТИФІКАЦІЇ ОБЛИЧЧЯ

У цьому розділі будуть описані можливі шляхи покращення аутентифікації за допомогою глибинного навчання та нейронних мереж, особливості роботи згорткових нейронних мереж. Також будуть розглянуті принципи побудови систем розпізнавання обличчя на основі штучного інтелекту і переваги та недоліки використання штучного інтелекту.

3.1 Покращення аутентифікації за допомогою глибинного навчання

Сучасні дослідники акцентують увагу на штучному інтелекті (AI) для подолання слабкостей та обмежень традиційних методів розпізнавання обличчя. Розвиток цих технологій відбувається за допомогою застосування досягнень таких підгалузей AI, як комп'ютерний зір, нейронні мережі та машинне навчання (ML) [7].

Відзначається помітний технологічний прорив у глибиному навчанні (DL). Глибоке навчання є частиною ML і ґрунтується на використанні штучних нейронних мереж.

Основна відмінність між DL та іншими методами машинного навчання полягає в навчанні представлення те, що таке навчання не потребує спеціалізованих алгоритмів для кожної конкретної задачі. Глибинне навчання грає важливу роль в системах розпізнавання обличчя, оскільки воно дозволяє автоматично вивчати складні візуальні ознаки та знаходити релевантні залежності у великих обсягах даних [13].

Використовуючи глибинне навчання, система розпізнавання обличчя стає стійкою до змін умов зйомки, таких як освітлення, позиція, масштаб та вираз обличчя. Глибинні нейронні мережі дозволяють побудувати моделі, які точно розрізняють різні обличчя, що є важливим для завдань ідентифікації та аутентифікації.

Завдяки здатності навчання на великих наборах даних, моделі глибинного навчання можуть досягати високої точності в розпізнаванні обличчя, зменшуючи кількість помилкових співпадінь та відмов у розпізнаванні.

3.1.1 Особливості роботи згорткових нейронних мереж

Прогрес DL в значній мірі обумовлений використанням згорткових нейронних мереж (CNN). CNN – це тип нейронної мережі, який спеціалізується на обробці зображень і використовує оператори згортки для автоматичного виявлення візуальних ознак у вхідних даних.

Основний принцип роботи CNN полягає в тому, що він використовує шари згортки, пулінгу і повнозв'язаних шарів для виявлення і ідентифікації різних характеристик зображень. Вхідні дані проходять через шар згортки, який застосовує фільтри для витягування різних візуальних ознак, таких як краї, форми, текстури і т.д. Потім застосовується шар пулінгу для зменшення розміру отриманого зображення і підвищення інваріантності до масштабу та позиції. На останньому етапі використовуються повнозв'язані шари для класифікації або виконання інших завдань на основі виявлених ознак [16].

Раніше, штучні нейронні мережі потребували величезних обчислювальних ресурсів для навчання та застосування повністю зв'язаних моделей з великою кількістю шарів штучних нейронів. З появою CNN цей недолік був подоланий. Крім того, в нейронних мережах, що використовуються у глибокому навчанні, є багато прихованих шарів нейронів. Сучасні методи DL дозволяють навчати і використовувати всі шари.

3.1.2 Способи покращення нейронних мереж

Нейронні мережі використовуються широко для розпізнавання обличчя, оскільки вони дозволяють автоматично вивчати складні візуальні ознаки та знаходити залежності у великих обсягах даних. Таблиця 3.1 демонструє різні способи покращення нейронних мереж для систем розпізнавання обличчя та надає опис кожного з цих способів [13].

Таблиця 3.1 – Способи покращення нейронних мереж

Спосіб	Опис
Knowledge distillation	Поєднання двох подібних мереж різних розмірів, де більша мережа навчає меншу. Після навчання менша мережа дає такий самий результат, як і велика, але робить це швидше.
Transfer learning	Сфокусоване на тренуванні всієї мережі або окремих шарів на конкретному наборі тренувальних даних. Це створює можливість усунути вузькі місця. Наприклад, ми можемо покращити точність, використовуючи набір зображень того самого типу, на якому найчастіше виникають помилки.
Quantization	Зменшення кількості обчислень і обсягу використовуваної пам'яті. Приближення плаваючої крапки низькобітовими числами допомагає в цьому.
Depthwise separable convolutions	Розробники створюють на основі таких шарів CNN, які мають менше параметрів і вимагають менше обчислень, але забезпечують хорошу продуктивність у розпізнаванні зображень, зокрема облич.

Важливо навчити глибоку згорткову нейронну мережу (DCNN) витягати унікальні обличчя зображень облич. Крім того, важливо надати DCNN здатність компенсувати вплив зсувів, різних кутів та інших спотворень на результат його розпізнавання [17]. Завдяки розширенню даних, зображення змінюються у всіх можливих способах перед навчанням. Це допомагає зменшити ризики, пов'язані з різними кутами, спотвореннями тощо. Чим більше різноманітності зображень використовується під час навчання, тим краще модель буде узагальнювати [6]. Забезпечення швидкого та безпомилкового розпізнавання автоматизованою системою є основним. У багатьох випадках це вимагає навчання системи з оптимальною швидкістю на

дуже великих наборах даних. Глибоке навчання допомагає знайти відповідь на цей виклик [7].

3.2 Принцип побудови систем розпізнавання обличчя на основі штучного інтелекту

Побудова систем розпізнавання обличчя на основі штучного інтелекту можлива двома способами:

1) розробка власної моделі;
2) використання готових моделей глибинного навчання для розпізнавання обличчя. Моделі, такі як DeepFace, FaceNet та інші, спеціально розроблені для завдань розпізнавання обличчя.

Для розробки власної моделі розпізнавання обличчя на основі штучного інтелекту необхідно виконати наступні кроки [18]:

– збір набору даних – потрібно зібрати достатню кількість зображень обличчя для тренування моделі, цей набір даних повинен включати різні особи з різних куточків зйомки та у різних умовах освітлення та позиції обличчя;
– анотування даних – кожне зібране зображення повинно бути анотоване, тобто позначене тегами або мітками, що вказують на особу на зображенні;
– передобробка даних – зібрані дані потрібно підготувати для використання у моделі, це може включати зміну розміру зображень, нормалізацію яскравості, вирівнювання облич та інші методи обробки даних;
– вибір архітектури моделі – наступним кроком є вибір архітектури моделі, яка буде використовуватись для розпізнавання обличчя, це може бути згорткова нейронна мережа (CNN) або комбінація різних типів мереж;
– тренування моделі – після вибору архітектури моделі необхідно навчити її розпізнавати обличчя, це включає передачу зображень

через модель, обчислення втрати та коригування ваг моделі для покращення результатів;

- тестування та налаштування моделі – після тренування модель потрібно протестувати на нових зображеннях, які раніше не використовувалися для тренування, це допоможе оцінити точність та ефективність моделі, при потребі модель можна налаштувати, внесенням змін до архітектури або параметрів моделі, для поліпшення результатів;
- розгортання моделі – після успішного тренування та тестування модель можна розгорнути на платформі або в додатку, де вона буде використовуватись для розпізнавання обличчя.

Важливо відзначити, що розробка власної моделі вимагає значних обчислювальних ресурсів, великої кількості даних та експертного розуміння алгоритмів машинного навчання.

3.3 Переваги та недоліки використання штучного інтелекту

Використання штучного інтелекту для систем розпізнавання обличчя має такі переваги:

1) штучний інтелект може досягати вражаючої точності в розпізнаванні обличчя, особливо з використанням глибинного навчання та згорткових нейронних мереж;
2) системи розпізнавання обличчя, побудовані на основі штучного інтелекту, можуть працювати автоматично без необхідності втручання людини, це забезпечує зручність та ефективність в різних ситуаціях, таких як контроль доступу або відеоспостереження [19];
3) штучний інтелект дозволяє легко масштабувати системи розпізнавання обличчя для обробки великої кількості даних та виконання розпізнавання в реальному часі.

Недоліки використання штучного інтелекту при розробці системи розпізнавання обличчя:

1) використання систем розпізнавання обличчя, особливо в громадських місцях або без належного контролю, може порушувати приватність людей, існує ризик збору та використання персональних даних без належного дозволу або контролю [19];
2) деякі системи розпізнавання обличчя можуть показувати низьку точність при розпізнаванні осіб з певних етнічних груп або при виявленні розбіжностей у розпізнаванні осіб різних расових груп, це може призводити до систематичних помилок та неправильного використання;
3) існує потенційний ризик зловживання системами розпізнавання обличчя, такими як масовий нагляд або стеження за особистими даними без належного контролю або дозволу.
4) якщо система розпізнавання обличчя не належно захищена, вона може стати предметом зламу або маніпуляцій, що може призвести до неправильного розпізнавання обличчя або навіть підробки.

Враховуючи ці переваги та недоліки, важливо розробляти та використовувати системи розпізнавання обличчя з урахуванням етичних, правових та приватних аспектів, забезпечуючи належний контроль, безпеку та захист приватності користувачів.

4 РОЗРОБКА ТА ТЕСТУВАННЯ LIVENESS DETECTION

У цьому розділі буде розібрано особливості налаштування середовища розробки та створення тестового набору даних. Також більш детально буде розглянуто деталі реалізації та тестування розробленого методу.

4.1 Способи визначення реального обличчя в кадрі

Для ефективного вирішення задач ідентифікації та аутентифікації користувача з використанням біометрії обличчя необхідно надійно виявляти наявність живої людини (розпізнавання реальної присутності) в кадрі – тобто повинна бути перевірка, що біометричний шаблон створюється саме на основі даних людини, а не, наприклад, на основі надрукованого зображення, піднесеного до камери. Для того, щоб зробити системи розпізнавання облич надійнішими, потрібно використовувати алгоритми, які здатні виявляти фальшиві обличчя.

Існує кілька способів визначення, чи є обличчя, яке розпізнається системою, реальним або підробленим:

1) аналіз текстури, включаючи обчислення локальних бінарних шаблонів (LBPs) на областях обличчя та використання методу опорних векторів (SVM) для класифікації облич як реальних чи підроблених [6];
2) аналіз частоти, такий як дослідження обличчя в області частоти Фур'є;
3) аналіз змінної фокусування, такий як дослідження варіації значень пікселів між двома послідовними кадрами;
4) алгоритми на основі евристичних методів, включаючи виявлення руху очей, руху губ та блимання, цей набір алгоритмів намагається відстежувати рух очей та блимання, щоб переконатися, що користувач не тримає фото іншої особи (оскільки на фото не блимаються очі та не рухаються губи);

5) алгоритми оптичного потоку, а саме дослідження різниць та властивостей оптичного потоку, створеного з тривимірних об'єктів та двовимірних площин [6];
6) 3D форма обличчя, подібна до того, що використовується в системі розпізнавання облич компанією Apple в iPhone, дозволяє системі розпізнавання облич розрізняти між реальними обличчями та друкованими фотографіями/зображеннями іншої особи [9];
7) комбінації вищезгаданих методів, що дозволяють інженеру системи розпізнавання облич вибирати ті моделі виявлення, які відповідають їхній конкретній задачі.

4.2 Налаштування середовища розробки та створення набору даних для тренування

Задача полягає в тому, щоб, отримавши вхідне зображення, навчити нейронну мережу, яка зможе розрізняти реальні обличчя від фальшивих/підроблених облич на екрані.

Розроблений алгоритм можна легко розширити на інші типи облич, що є підробленими, включаючи друковані копії, високорозширені друковані зображення та інше.

Для набіру даних для тренування, було створено:

1) 25 секундне відео себе (знятому в режимі селфі/портрету);
2) запис на камеру комп’ютера або ноутбуку, увімкнувши вже записане відео;
3) в результаті отримуємо два відео – одне з реальним обличчям, інше -з підробленим обличчям.

Задля реалізації алгоритму використовували TensorFlow. TensorFlow – це відкрите програмне забезпечення для машинного навчання та штучного інтелекту, розроблене компанією Google Brain Team та випущене у 2015 році [20]. Він дозволяє розробникам створювати, навчати та розгортати моделі машинного навчання для розв'язання різноманітних завдань, включаючи класифікацію, регресію, обробку зображень, обробку мовлення та інші.

TensorFlow має багато інструментів та бібліотек для спрощення процесу розробки та навчання моделей машинного навчання, включаючи Keras, TensorFlow Lite, TensorFlow.js та TensorBoard. TensorFlow може працювати на різних платформах, включаючи комп'ютери, сервери, мобільні пристрої та IoT-пристрої.

TensorFlow використовує графову модель обчислень, що дозволяє ефективно виконувати обчислення на графічних процесорах (GPU) та інших апаратних пристроях з високою продуктивністю.

4.3 Створення архітектурного рішення

На рисунку 4.1, що представлений нижче зображено структуру проекту, вона складається з 4 основних директорій:

1) dataset/ – в цій директорії зберігається два класи зображень: підроблені зображення, зняті камерою, спрямованою на екран під час відтворення відео мого обличчя; реальні зображення мене, зняті з відео-селфі за допомогою мого телефону;
2) face_detector/ – у цьому каталозі знаходиться попередньо навчена модель обличчя, створена на основі фреймворку Caffe, яка дозволяє виділяти окремі обличчя з загального зображення облич;
3) pyimagesearch/ – цей модуль містить наш клас LivenessNet;
4) videos/ – ця директорія, в свою чергу, зберігає два вхідних відео для навчання нашого класифікатора LivenessNet.

```
(base) sofiiahalamai@Sofiias-MacBook-Pro liveness-detection % tree --dirsfirst --filelimit 100
.
├── dataset
│   ├── fake  [322 entries exceeds filelimit, not opening dir]
│   └── real  [402 entries exceeds filelimit, not opening dir]
├── face_detector
│   ├── deploy.prototxt
│   └── res10_300x300_ssd_iter_140000.caffemodel
├── liveness.model
│   ├── assets
│   ├── variables
│   │   ├── variables.data-00000-of-00001
│   │   └── variables.index
│   ├── fingerprint.pb
│   ├── keras_metadata.pb
│   └── saved_model.pb
├── pyimagesearch
│   ├── __init__.py
│   └── livenessnet.py
├── videos
│   ├── fake.mp4
│   └── real.MOV
├── fake.mp4
├── gather_examples.py
├── le.pickle
├── liveness_demo.py
├── plot.png
├── real.mov
└── train.py

10 directories, 18 files
```

Рисунок 4.1 – Структура проекту

В проекті також створено три скрипти Python:

1) скрипт gather_examples.py призначений для отримання зображень облич (ROIs відносяться до облич, тобто визначають окремі ділянки зображення, які містять обличчя людей, які будуть використані для тренування моделі виявлення реального) з вхідних відеофайлів та допомагає створити набір даних для глибинного навчання;
2) train.py – цей скрипт призначений для навчання класифікатора LivenessNet. Для цього використовувалось Keras та TensorFlow для навчання моделі. Процес навчання призведе до створення кількох файлів:
    a. le.pickle: це файл, що містить шифратор міток класу;
    b. liveness.model: це файл, що містить серіалізовану модель Keras, яка використовується для виявлення реального обличчя;
    c. plot.png: це графік історії навчання, що показує криві точності та втрат моделі, який дозволяє оцінити її ефективність та визначити, чи є ознаки перенавчання або недонавчання.

3) liveness_demo.py – це демонстраційний скрипт, який дозволяє вам запустити веб-камеру і в режимі реального часу виявляти реальне обличчя на кадрах.

4.4 Виявлення та вилучення областей обличчя

Мета скрипту gather_examples.py є наповнення двох каталогів – кадрами на рисунку 4.2 зображено умовну роботи скрипту. Кадри потім будуть використовуватися для навчання системи глибинного навчання для виявлення реального обличчя:

- dataset/fake/: містить обличчя ROIs з файлу fake.mp4;
- dataset/real/: містить обличчя ROIs з файлу real.mov.

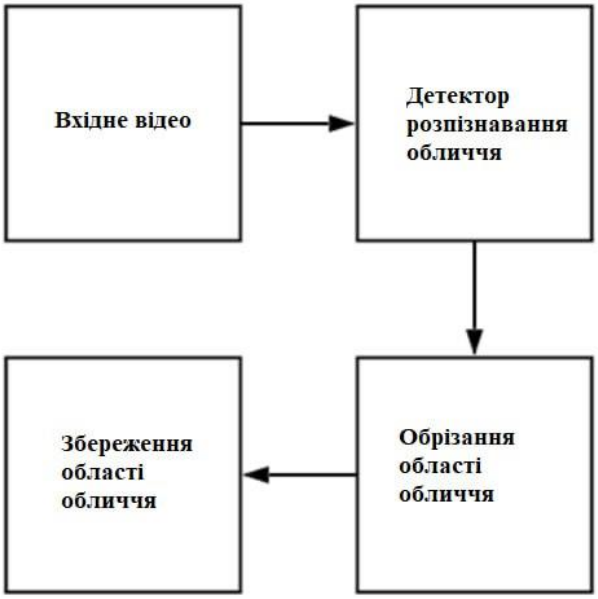

Рисунок 4.2 - Виявлення областей облич на відео для створення набору даних

Для створення скрипту потрібні лише OpenCV [5] (відкрите програмне забезпечення з відкритим кодом, призначене для обробки зображень та комп'ютерного зору) та NumPy [21] (це бібліотека мови програмування Python для обробки числових даних), крім вбудованих модулів Python. Скрипт базується на припущенні, що в кожному кадрі відео є тільки одне обличчя. Вихідний код цього скрипту представлено в Лістингу А.1.

Також до заданого скрипту додано можливість вводити аргументи з командного рядка, такі як:

- --input: шлях до вхідного відеофайлу;
- --output: шлях до каталогу виведення, куди будуть зберігатись вирізані обличчя;
- --detector: шлях до детектора облич OpenCV, який використовує глибинне навчання;
- --confidence: мінімальна ймовірність для фільтрації слабких виявлень облич, за замовчуванням це значення становить 50%;
- --skip: не потрібно виявляти та зберігати кожне зображення, оскільки сусідні кадри будуть подібні, пропускання N кадрів, за допомогою цього аргументу можна змінити значення за замовчуванням, яке дорівнює 16.

Для виконання скрипту gather_examples.py, щоб виділити обличчя для “fake” класу (результат виконання представлено на Рисунку 4.3), необхідно виконати наступну команду:

```
liveness-detection % python gather_examples.py -i
videos/fake_video.mp4 -o dataset/fake \-detector
human_face_detector -s 5
```

```
(base) sofiiahalamai@Sofiias-MacBook-Pro liveness-detection % python gather_examples.py --input videos/fake.mp4 --output dataset/fake
\--detector face_detector --skip 3
[INFO] loading face detector...
[INFO] saved dataset/fake/0.png to disk
[INFO] saved dataset/fake/1.png to disk
[INFO] saved dataset/fake/2.png to disk
[INFO] saved dataset/fake/3.png to disk
[INFO] saved dataset/fake/4.png to disk
[INFO] saved dataset/fake/5.png to disk
[INFO] saved dataset/fake/6.png to disk
[INFO] saved dataset/fake/7.png to disk
[INFO] saved dataset/fake/8.png to disk
[INFO] saved dataset/fake/9.png to disk
[INFO] saved dataset/fake/10.png to disk
[INFO] saved dataset/fake/11.png to disk
[INFO] saved dataset/fake/12.png to disk
[INFO] saved dataset/fake/13.png to disk
[INFO] saved dataset/fake/14.png to disk
[INFO] saved dataset/fake/15.png to disk
[INFO] saved dataset/fake/16.png to disk
[INFO] saved dataset/fake/17.png to disk
[INFO] saved dataset/fake/18.png to disk
```

Рисунок 4.3 – Результат виконання скрипту gather_examples.py для “fake” класу

Аналогічне виконання для “real” класу, результат представлено на Рисунку 4.4.

```
(base) sofiiahalamai@Sofiias-MacBook-Pro liveness-detection % python gather_examples.py --input videos/real.mov --output dataset/real
\--detector face_detector --skip 4
[INFO] loading face detector...
[INFO] saved dataset/real/0.png to disk
[INFO] saved dataset/real/1.png to disk
[INFO] saved dataset/real/2.png to disk
[INFO] saved dataset/real/3.png to disk
[INFO] saved dataset/real/4.png to disk
[INFO] saved dataset/real/5.png to disk
[INFO] saved dataset/real/6.png to disk
[INFO] saved dataset/real/7.png to disk
[INFO] saved dataset/real/8.png to disk
[INFO] saved dataset/real/9.png to disk
[INFO] saved dataset/real/10.png to disk
[INFO] saved dataset/real/11.png to disk
[INFO] saved dataset/real/12.png to disk
[INFO] saved dataset/real/13.png to disk
[INFO] saved dataset/real/14.png to disk
[INFO] saved dataset/real/15.png to disk
[INFO] saved dataset/real/16.png to disk
[INFO] saved dataset/real/17.png to disk
[INFO] saved dataset/real/18.png to disk
[INFO] saved dataset/real/19.png to disk
[INFO] saved dataset/real/20.png to disk
[INFO] saved dataset/real/21.png to disk
```

Рисунок 4.4 – Результат виконання скрипту gather_examples.py для “real” класу

Після виконання скриптів отримуємо наступну кількість зображень:

- fake: 322 зображень;
- real: 402 зображень;
- всього: 311 зображень.

4.5 Реалізація "LivenessNet" детектора на глибинному навчанні

В основі LivenessNet (див. Рисунок 3.5) насправді проста згорткова нейронна мережа. Мережа буде максимально поверхневою та з якомога меншою кількістю параметрів з двох причин: щоб зменшити шанси перенавчання на нашому невеликому наборі даних; щоб забезпечити можливість роботи детектора в реальному часі.

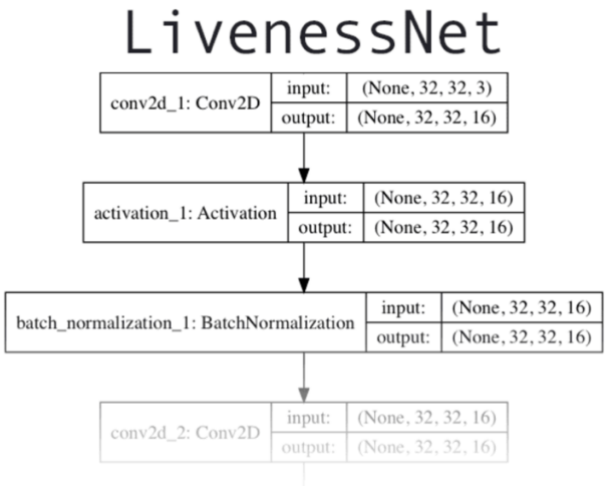

Рисунок 4.5 – Архітектура глибиного навчання для LivenessNet

У класі LivenessNet, скрипт livenessnet.py (див. Лістинг А.2), є лише один статичний метод під назвою "build", який приймає чотири параметри:

- width – ширина зображення/об'єму;
- height – висота зображення;
- depth – кількість каналів для зображення, у випадку RGB зображень це буде 3;
- classes – кількість класів, які включають у себе "real" та "fake" класи.

Створена згорткова нейронна мережа має якості, схожі на VGGNet (глибока згорткова нейронна мережа, яка була розроблена з метою розпізнавання образів в базі даних ImageNet). Вона є дуже малодійною і містить лише кілька навчених фільтрів.

Спочатку створюється послідовність шарів, що складається з згорткового шару (CONV), функції активації (RELU), ще одного CONV, ще однієї функції активації та згорткового шару з підвибіркою (POOL), до якої додано нормалізацію пакетів та випадкове вимикання.

Потім до цієї послідовності додається ще один набір шарів, що складається з згорткового шару, RELU, ще одного згорткового шару, ще однієї RELU та POOL [22].

В кінці кінців, додаємо повнозв’язані шари (FC) з функцією активації RELU до цієї послідовності шарів.

4.5.1 Створення скрипту тренування детектора

Використовуючи як набір даних "real" і "fake" зображення, можемо тренувати модель виявлення реального обличчя за допомогою OpenCV, Keras [23] (високорівнева бібліотека машинного навчання, яка дозволяє легко і швидко створювати та тренувати нейронні мережі) та глибинного навчання, процес тренування зображено на Рисунку 4.6.

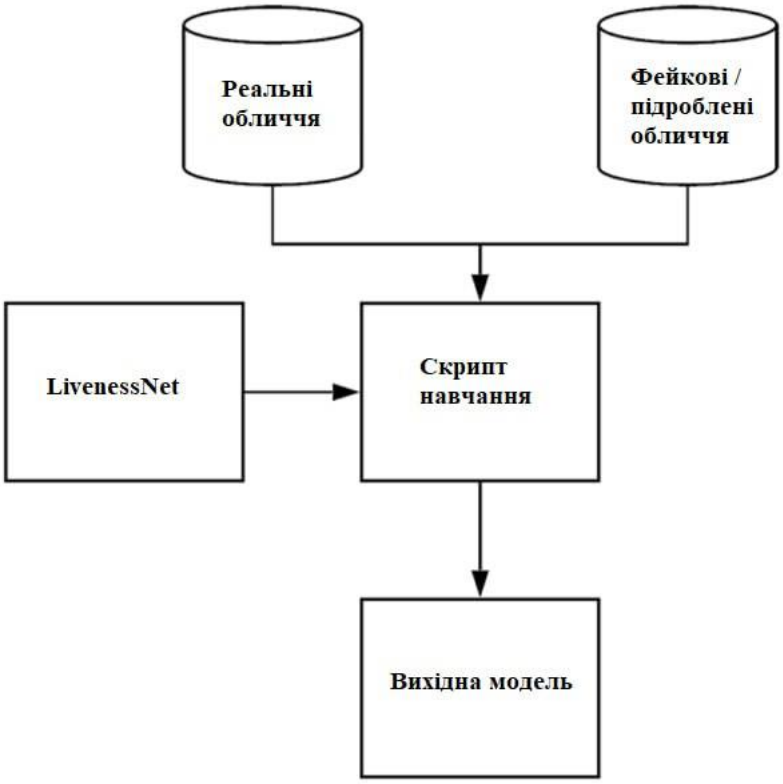

Рисунок 4.6 – Процес тренування LivenessNet

Враховуючи наявність набору даних з реальних/фальшивих зображень та реалізації LivenessNet, можна починати тренувати мережу, для цього необхідно створити файл train.py.

Скрипт тренування моделі (Лістинг А.3) для визначення реальності обличчя включає декілька імпортів:

1) matplotlib: використовується для створення графіку тренування;
2) LivenessNet: CNN для визначення реальної присутності, яку було визначено раніше;
3) train_test_split: функція з бібліотеки scikit-learn [24], яка дозволяє розбити дані на тренувальний та тестовий набори;
4) classification_report: також з scikit-learn, цей інструмент згенерує короткий статистичний звіт про продуктивність моделі;
5) ImageDataGenerator: використовується для збільшення обсягу набору даних шляхом створення випадкових змінених зображень;
6) pyplot: використовується для створення красивих графіків тренування;

7) numpy: бібліотека числової обробки даних для Python. Вона також є вимогою для OpenCV.

Скрипт train.py очікує отримати чотири параметри з командного рядка:

- --dataset: шлях до вхідного набору даних, який було створено раніше за допомогою скрипту gather_examples.py;
- --model: шлях до файлу, в який скрипт збереже вихідну модель;
- --le: шлях до файлу, в який скрипт збереже серіалізований шифратор міток;
- --plot: скрипт навчання буде генерувати графік.

Також до скрипту встановлюються параметри навчання, включаючи початкову швидкість навчання та розмір пакету. Дані складаються з зображень, які завантажуються та змінюють свій розмір на 32×32 пікселів. Кожне зображення має відповідну мітку, яка зберігається у списку міток. Всі інтенсивності пікселів масштабуються до діапазону [0, 1], після чого список перетворюється у масив NumPy [21].

Бібліотека scikit-learn [24] використовується для розділення даних – 75% використовується для тренування, а 25% залишається для тестування. Далі створюється об'єкт для нарощування даних, який здійснює зміну зображень шляхом випадкових обертів, масштабування, зсувів, зміщень та відображень. Після цього розпочинається процес навчання моделі, а після завершення можна оцінити її результати та створити графік тренування.

4.5.2 Навчання Liveness Detector

Для запуску скрипту тренування train.py, необхідно виконати наступну команду (процес та результат навчання представлено на Рисунку 3.7):

```
python train.py -d dataset -m liveness.detector.model -l
le.pickle
```

```
Epoch 42/50
67/67 [==============================] - 1s 18ms/step - loss: 0.0121 - accuracy: 0.9963 - val_loss: 0.0036 - val_accuracy: 1.0000
Epoch 43/50
67/67 [==============================] - 1s 18ms/step - loss: 0.0139 - accuracy: 0.9981 - val_loss: 0.0585 - val_accuracy: 1.0000
Epoch 44/50
67/67 [==============================] - 1s 17ms/step - loss: 0.0101 - accuracy: 1.0000 - val_loss: 0.0025 - val_accuracy: 1.0000
Epoch 45/50
67/67 [==============================] - 1s 18ms/step - loss: 0.0098 - accuracy: 0.9981 - val_loss: 0.0079 - val_accuracy: 1.0000
Epoch 46/50
67/67 [==============================] - 1s 17ms/step - loss: 0.0389 - accuracy: 0.9869 - val_loss: 0.0130 - val_accuracy: 1.0000
Epoch 47/50
67/67 [==============================] - 1s 18ms/step - loss: 0.0170 - accuracy: 0.9963 - val_loss: 0.0026 - val_accuracy: 1.0000
Epoch 48/50
67/67 [==============================] - 1s 21ms/step - loss: 0.0348 - accuracy: 0.9907 - val_loss: 0.0037 - val_accuracy: 1.0000
Epoch 49/50
67/67 [==============================] - 1s 18ms/step - loss: 0.0078 - accuracy: 1.0000 - val_loss: 0.0033 - val_accuracy: 1.0000
Epoch 50/50
67/67 [==============================] - 1s 17ms/step - loss: 0.0683 - accuracy: 0.9757 - val_loss: 0.0819 - val_accuracy: 0.9945
[INFO] evaluating network...
23/23 [==============================] - 0s 12ms/step
              precision    recall  f1-score   support

        fake       0.99      1.00      0.99        70
        real       1.00      0.99      1.00       111

    accuracy                           0.99       181
   macro avg       0.99      1.00      0.99       181
weighted avg       0.99      0.99      0.99       181

[INFO] serializing network to 'liveness.model'...
```

Рисунок 4.7 – Процес та результат навчання моделі

На Рисунку 4.8 представлено графік процесу навчання моделі, в результаті було отримано 100% точність розпізнавання реальної обличчя вказаного валідаційного набору даних з обмеженим перенавчанням.

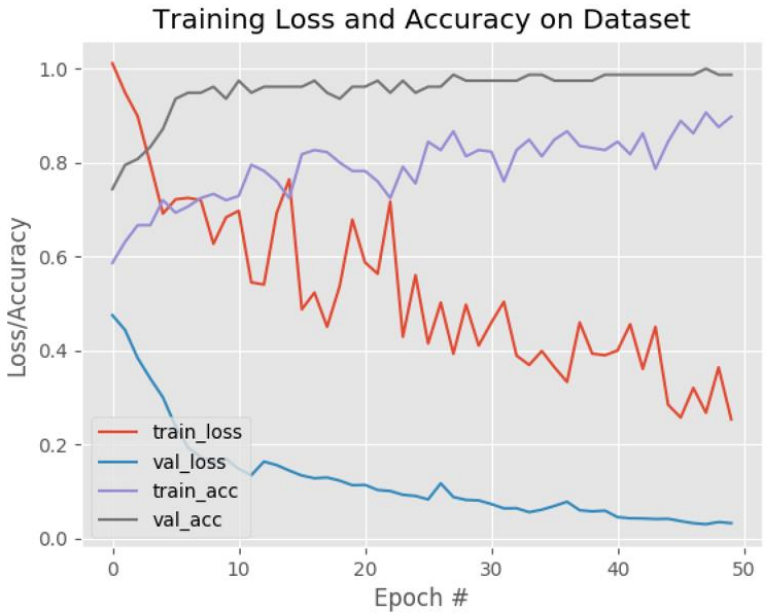

Рисунок 4.8 – Графік навчання моделі розпізнавання реального обличчя з використанням OpenCV, Keras та глибиного навчання

4.6 Розгортання детектора та тестування в режимі реального часу

Останнім кроком є поєднання всіх частин, для цього створено скрипт liveness_demo.py (вихідний код представлено в Лістингу А.4). Для розпізнавання обличчя створено цикл, всередині якого є:

- фільтрування слабких розпізнавань;

- вилучення координат меж обмежуючого прямокутника облича та переконуємося, що вони не виходять за межі кадру;
- вилучення ROI обличчя та попередньо обробка його таким же способом, як і навчальні дані;
- використовуємо детектор для визначення, чи є обличе реальним чи підробленим;
- зображення тексту мітки та прямокутник навколо обличчя з результатом.

Щоб долучитись до демонстрації розпізнавання реального обличчя необхідно виконати наступну команду:

```
python liveness_demo.py -m liveness.detector.model -l
le.pickle \-d human_face_detector
```

На Рисунку 4.9 можна побачити процес розгортання детектора розпізнавання реального обличчя для його тестування в реальному часі. Нижче наведено різні умови тестування, які включають:

1) присутність лише реальної особи;
2) зображення обличчя іншої людини на екрані іншого пристрою;
3) одночасна поява реальної особи та підробленого обличчя іншої людини;
4) присутність лише обличчя тієї ж самої людини, яка була раніше протестована як реальна, але зображена на екрані іншого пристрою;
5) одночасна поява реальної особи та підробленого обличчя тієї ж самої людини.

Для кожної з цих умов результати тестування відповідно наведено на Рисунках 4.10 - 4.14.

```
(base) sofiiahalamai@Sofiias-MacBook-Pro liveness-detection % python liveness_demo.py --model liveness.model --le le.pickle \
        --detector face_detector
[INFO] loading face detector...
[INFO] loading liveness detector...
Metal device set to: Apple M1

systemMemory: 16.00 GB
maxCacheSize: 5.33 GB

[INFO] starting video stream...
```

Рисунок 4.9 – Процес розгортання детектора розпізнавання реального обличчя

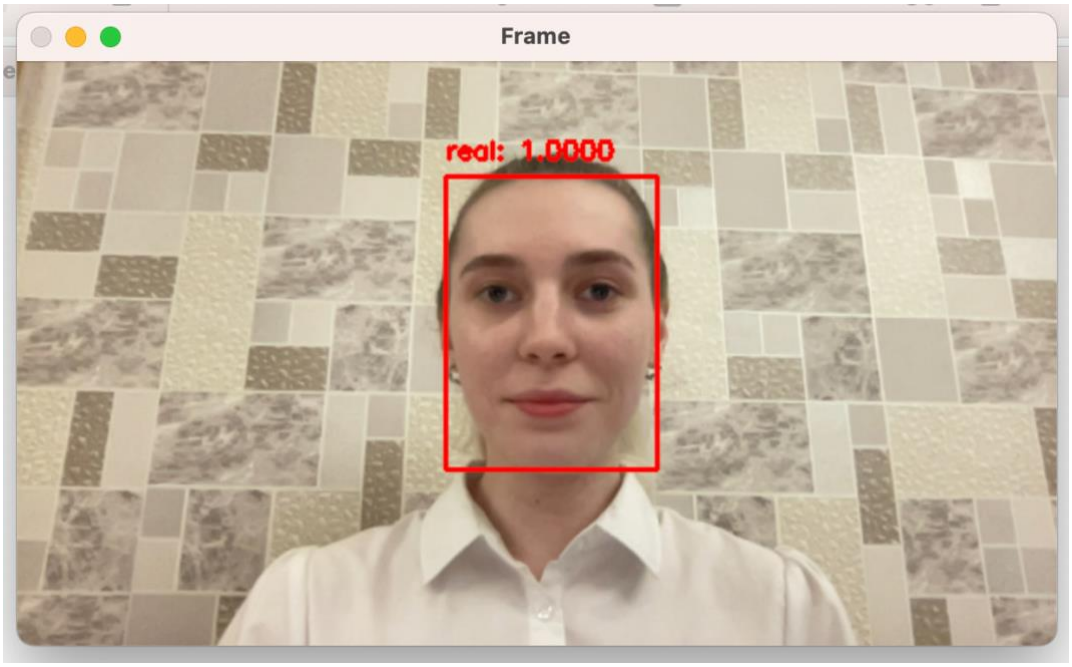

Рисунок 4.10 – Результат тестування при наявності в кадрі лише реальної особи

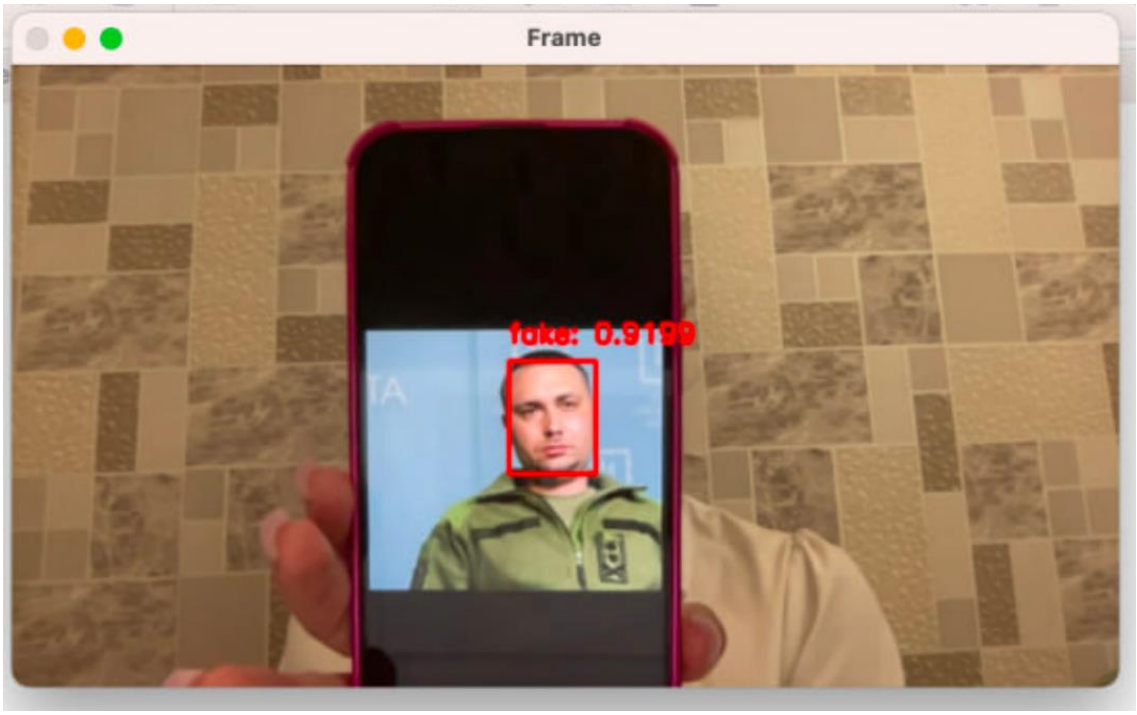

Рисунок 4.11– Результат тестування при наявності в кадрі іншої людини зображеної на екрані іншого пристрою

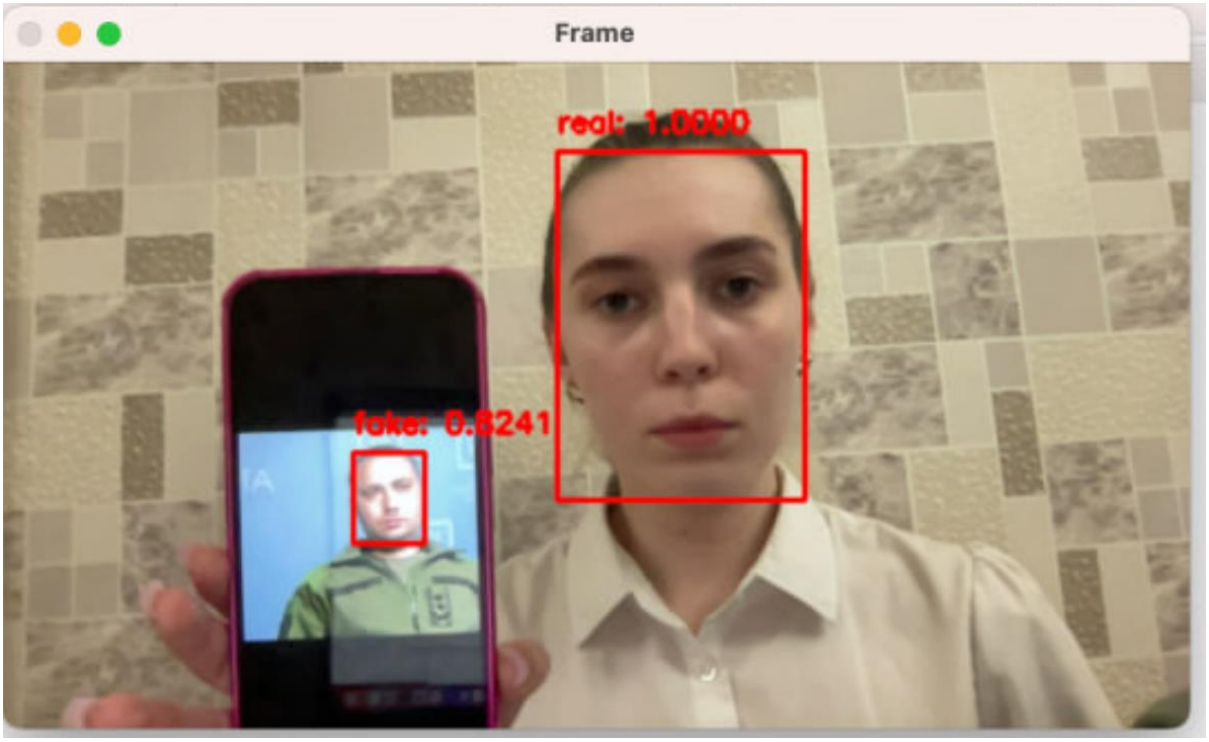

Рисунок 4.12 – Результат тестування при наявності в кадрі одночасної появи реальної особи та підробленого обличчя іншої людини

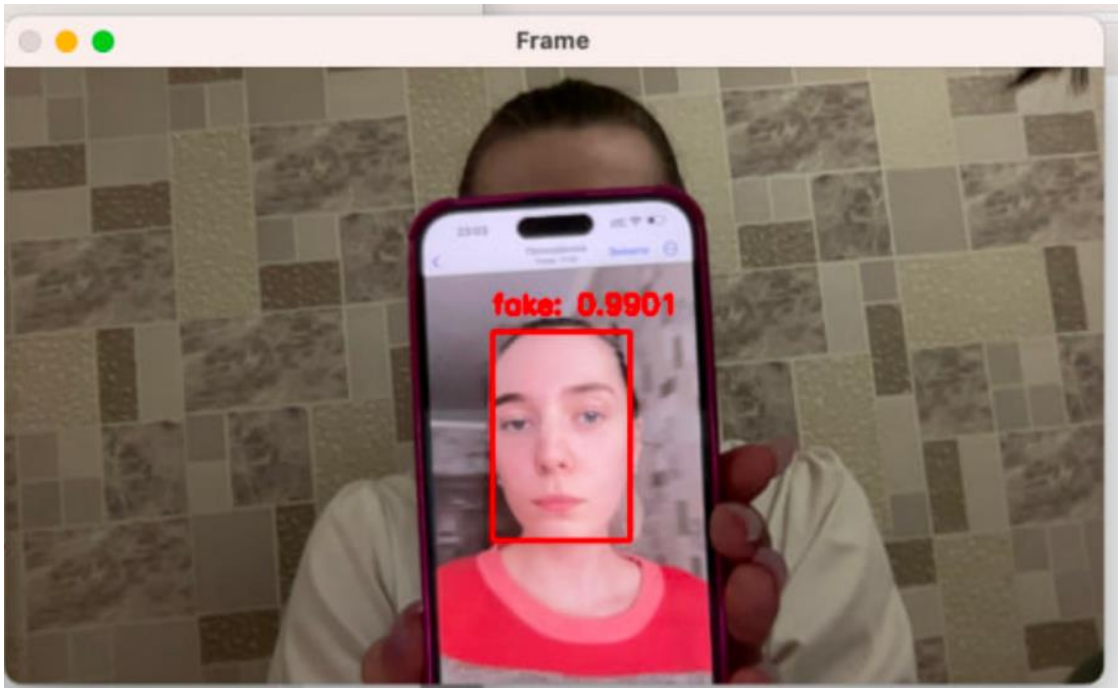

Рисунок 4.13 – Результат тестування при наявності в кадрі лише обличчя тієї ж самої людини, що була до цього протестована як реальна, але зображена на екрані іншого пристрою

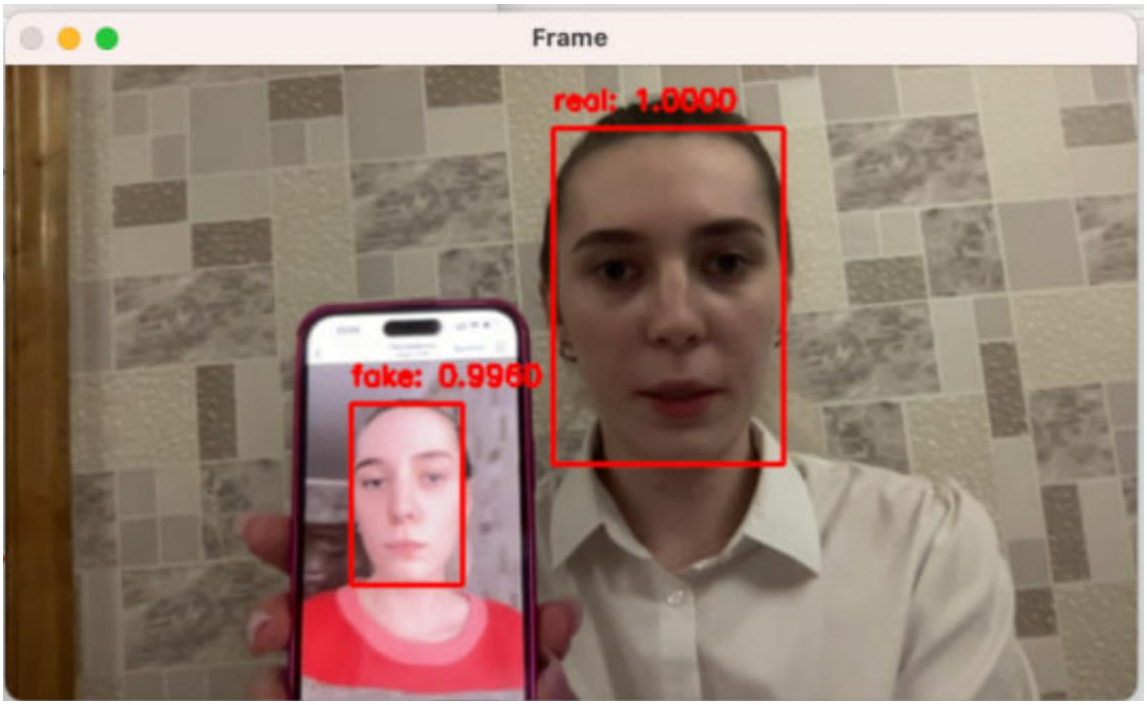

Рисунок 4.14 – Результат тестування при наявності в кадрі одночасної появи реальної особи та підробленого обличчя тієї ж самої людини

Далі оновлюємо набір даних для тренування, додавши два нові відео. Перше відео містить реальне обличчя нової людини в окуляри та без окулярів (нехай, умовно це буде особа-2), а друге відео містить підроблене обличчя цієї ж людини. Процес тренування системи глибокого навчання для виявлення реального обличчя нової людини проводимо аналогічним чином, як у початковому навчанні з особою-1.

Після завершення тренування, ми проводимо тестування для перевірки коректності аутентифікації. Тести включають наступні сценарії та відповідно результати тестування зображено на рисунках 4.15-4.17:

1) тест наявності в кадрі лише реальної особи-2 в окулярах;
2) тест на одночасну появу реальної особи-1 і особи-2 без окулярів;
3) тест на одночасну появу реальної особи-1 і підробленого обличчя особи-2 в більш гіршому освітленні ніж раніше використовувалось для тестування.

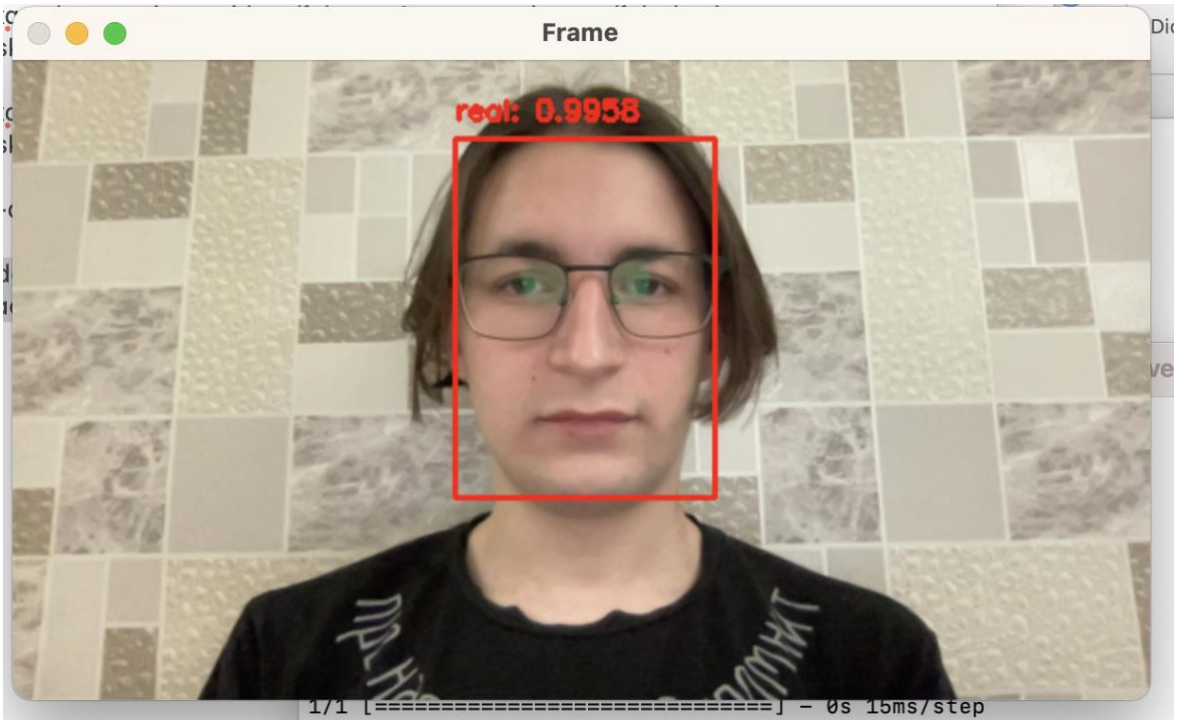

Рисунок 4.15 – Результат тестування при наявності в кадрі лише реальної особи-2 в окулярах

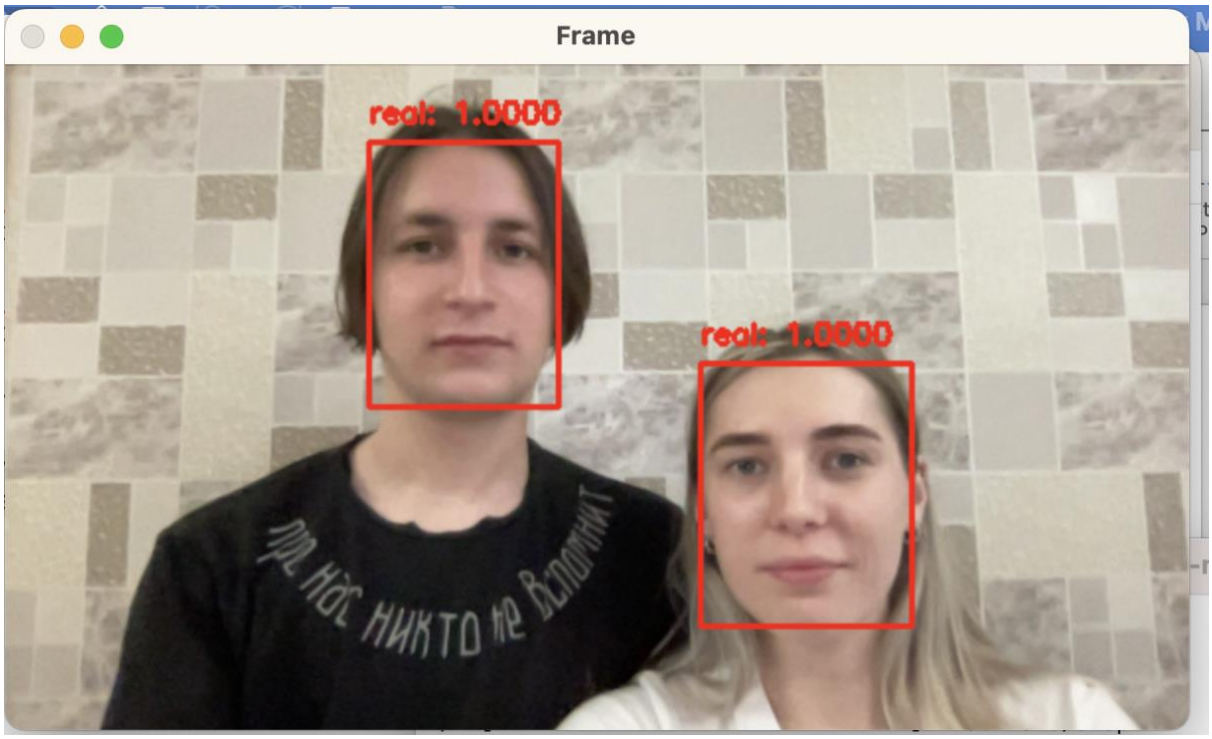

Рисунок 4.16 – Результат тестування при одночасній появі реальної особи-1 і особи-2 без окулярів

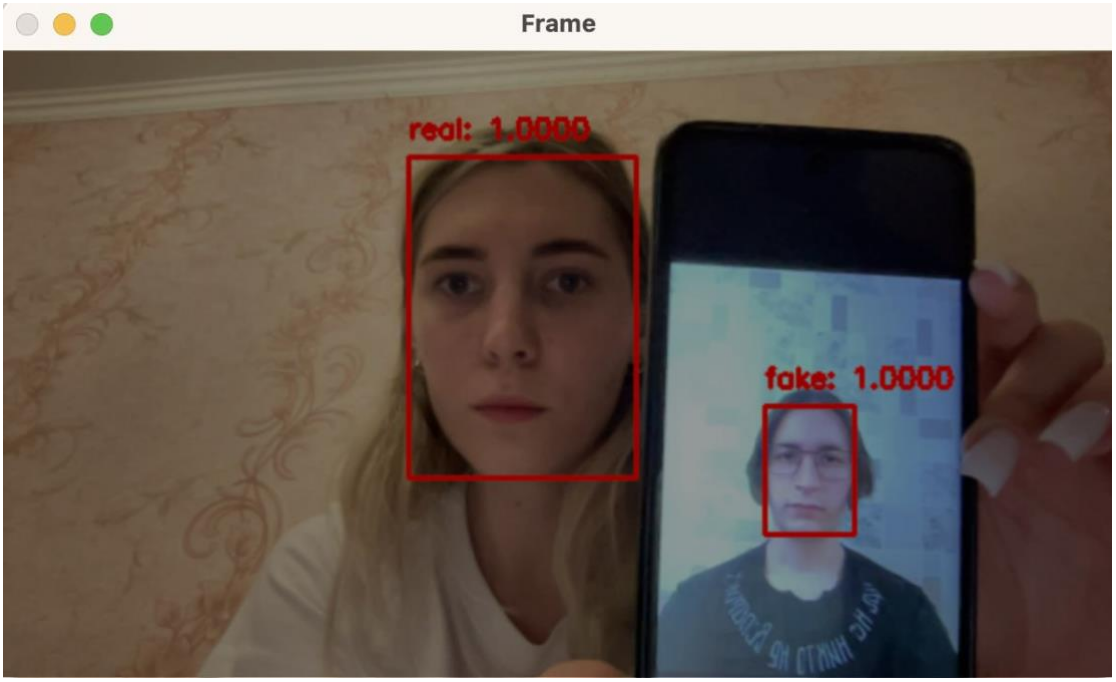

Рисунок 4.17 – Результат тестування при одночасній появі реальної особи-1 і підробленого обличчя особи-2 в більш гіршому освітленні

Далі додаємо ще 8 нових облич до тренувальної моделі. Процес тренування системи глибокого навчання для розпізнавання облич людей проводимо аналогічним чином. Після завершення тренування, побудуємо матрицю конфузії [25], таблиця 4.1, яка дозволяє зрозуміти, як часто модель правильно або неправильно класифікує обличчя в кожному класі.

Таблиця 4.1 – Матриця конфузії

Передбачено	Реальне обличчя	Підроблене обличчя
Реальне обличчя	9	1
Підроблене обличчя	0	10

З 10 справжніх облич, 9 були правильно класифіковані (TP), що свідчить про високу точність моделі в розпізнаванні справжніх облич.

Однак, випадок, коли 1 справжнє обличчя було помилково визнане підробленим (FN), вказує на наявність помилок в роботі моделі, коли реальна особа не була впізнана.

З 10 підроблених облич, всі вони були правильно класифіковані (TN), що свідчить про ефективність моделі в розпізнаванні підроблених облич.

У даному випадку не було помилково класифікованих підроблених облич (FP), що є позитивним результатом.

Загалом, модель має високу точність в розпізнаванні як справжніх, так і підроблених облич, але оскільки вибірка була малою, це може вплинути на достовірність оцінок ефективності моделі, які базуються на матриці конфузії. Однак, варто звернути увагу на наявність помилок при класифікації справжніх облич, оскільки неправильне визнання реальної особи як підробленої може мати серйозні наслідки з погляду безпеки та автентифікації. Таким чином, модель може потребувати подальшої оптимізації та вдосконалення, щоб зменшити кількість помилок і забезпечити більш точну класифікацію облич.

Підводячи підсумок з результатів тестування, можна стверджувати, що всі вищевказані тести були успішно пройдені. Детектор розпізнавання реального обличчя добре функціонує при присутності реальних облич, і може ефективно розрізняти їх від підроблених. Навіть в складних умовах, таких як одночасна поява реальної особи та підробленого обличчя, при погано освітлені та за наявності окулярів, детектор розпізнавання реального обличчя продемонстрував надійність та точність. Чим більше даних було надано для тестування, тим точнішим був детектор. Такі результати тестування дають підстави для впевненості в ефективності детектора та його можливості використовуватися в різних сферах, де необхідна біометрична аутентифікація на основі розпізнавання облич.

ВИСНОВКИ

Метою даної роботи було розробити ефективний метод біометричної аутентифікації, який базується на розпізнаванні облич та поєднує високу точність та швидкість ідентифікації. Особливої уваги було надано важливості відповідному розрізненню між живим обличчям та цифровими зображеннями або відео, які можуть використовуватися для обхідного доступу до системи.

Також надати розуміння сучасного положення справ щодо систем розпізнавання обличчя, та технологій, що використовуються при реалізації даного методу.

Для досягнення даної мети були вирішені декілька завдань.

Перш за все було розглянуті основні поняття для ознайомлення з предметною областю, та визначення основних методів та типів біометричної аутентифікації.

Були розглянуті основні аспекти створення систем розпізнавання обличчя, принципи їх роботи, характеристика 2D, 3D та технологій аналітики обличчя.

Був проведений детальний розгляд недоліків та обмежень традиційного підходу розпізнавання, та розглянуті найпопулярніші на даний момент рішення такі як: Apple Face ID, Facebook DeepFace, Microsoft Face API.

Були проведені експерименти на реалізованому Liveness Detection. Зведені результати тестування свідчать про успішне пройдення всіх зазначених тестів. Детектор розпізнавання облич демонструє високу ефективність при виявленні реальних осіб і ефективно відрізняє їх від підроблених. Навіть в умовах, коли одночасно присутня реальна особа і підроблене обличчя, при поганому освітленні та за наявності окулярів, детектор розпізнавання реальних облич проявив надійність та точність. Ці результати тестування підтверджують ефективність детектора та підтримують його використання в різних галузях, де потрібна біометрична аутентифікація

на основі розпізнавання облич. З побудованої матриці конфузії можна зробити такий висновок щодо подальшого покращення створеного методу, а саме модель потребує подальшої оптимізації та вдосконалення, щоб зменшити кількість помилок і забезпечити більш точну класифікацію облич.

У цілому, сучасна технологія розпізнавання обличчя прогресує і має значний потенціал у багатьох сферах. Проте, необхідно уважно розглядати питання приватності та етики, а також продовжувати дослідження та розробки для подальшого вдосконалення цієї технології.

ПЕРЕЛІК ВИКОРСТАНИХ ДЖЕРЕЛ

1. Thales Group. "Digital Identity Solutions." URL: https://www.thalesgroup.com/en/markets/digital-identity-and-security/government/inspired/biometrics (дата звернення 04.03.2023)
2. TechTarget. "Authentication." URL: https://www.techtarget.com/searchsecurity/definition/authentication (дата звернення 04.03.2023)
3. IBM. "Computer Vision." URL: https://www.ibm.com/topics/computer-vision (дата звернення 13.03.2023)
4. Apple Support. "About Touch ID advanced security technology." URL: https://support.apple.com/en-gb/HT208108 (дата звернення 15.03.2023)
5. OpenCV Documentation. "Tutorials." URL: https://docs.opencv.org/2.4/doc/tutorials/tutorials.html (дата звернення 02.04.2023)
6. Stan Z. Li та Anil K. Jain. Handbook of Face Recognition – Springer, 2011 – 141 с.
7. Mark Andrejevic та Neil Selwyn. Facial Recognition 1st Edition – Polity 2022, 72 с.
8. Faceter. "2D and 3D Face Recognition Technologies: Pros and Cons." URL: https://faceter.cam/en/blog/2d-and-3d-face-recognition-technologies-pros-and-cons/ (дата звернення 16.03.2023)
9. SpeechOcean. "Text-to-Speech for the Welsh language." URL: https://en.speechocean.com/Cy/109.html (дата звернення 20.04.2023)
10. Facebook Research. "DeepFace: closing the gap to human-level Performance in verification of face." URL: https://research.facebook.com/publications/deepface-closing-the-gap-to-human-level-performance-in-verification/ (дата звернення 22.04.2023)

11. Microsoft Azure. "Cognitive Services - Face API." URL: https://azure.microsoft.com/en-gb/products/cognitive-services/face (дата звернення 16.04.2023)
12. FastDataScience. "Building a Face Recognizer." URL: https://fastdatascience.com/building-a-face-recogniser/ (дата звернення 22.04.2023)
13. Mobidev. "Improve AI Facial Recognition Accuracy with Machine & Deep Learning." URL: https://mobidev.biz/blog/improve-ai-facial-recognition-accuracy-with-machine-deep-learning (дата звернення 22.04.2023)
14. Global Market Insights. "Facial Recognition Market Size By Component" URL: https://www.gminsights.com/industry-analysis/facial-recognition-market (дата звернення 26.04.2023)
15. Allied Market Research. "Facial Recognition Market Statistics: 2030" URL: https://www.alliedmarketresearch.com/facial-recognition-market (дата звернення 26.04.2023)
16. IBM. "Convolutional Neural Networks." URL: https://www.ibm.com/topics/convolutional-neural-networks (дата звернення 28.04.2023)
17. Packt. "Deep Convolutional Neural Network (DCNN)." URL: https://subscription.packtpub.com/book/big-data-and-business-intelligence/9781787128422/3/ch03lvl1sec24/deep-convolutional-neural-network-dcnn (дата звернення 29.04.2023)
18. Towards Data Science. "How to Build a Face Detection and Recognition System." URL: https://towardsdatascience.com/how-to-build-a-face-detection-and-recognition-system-f5c2cdfbeb8c (дата звернення 29.04.2023)
19. GeeksforGeeks. "Face Recognition using Artificial Intelligence." URL: https://www.geeksforgeeks.org/face-recognition-using-artificial-intelligence/ (дата звернення 07.05.2023)
20. TensorFlow Documentation. "TensorFlow." URL: https://www.tensorflow.org/learn (дата звернення 10.05.2023)

21. NumPy Documentation. "NumPy." URL: https://numpy.org/doc/stable/ (дата звернення 11.05.2023)
22. Baeldung. "Machine Learning - RELU, Dropout Layers." URL: https://www.baeldung.com/cs/ml-relu-dropout-layers (дата звернення 13.05.2023)
23. Keras Documentation. "Keras." URL: https://keras.io/api/ (дата звернення 15.05.2023)
24. scikit-learn Documentation. "scikit-learn User Guide." URL: https://scikit-learn.org/stable/user_guide.html (дата звернення 17.05.2023)
25. Intechopen. “Face Recognition: Demystification of Multifarious Aspect in Evaluation Metrics” URL: https://www.intechopen.com/chapters/50437# (дата звернення 18.05.2023)

ДОДАТОК А

Лістинги вихідного коду Liveness Detection: розпізнавання реального обличчя в кадрі.

Лістинг А.1 – Скрипт gather_examples.py

```
import cv2
import numpy as np
import os
import argparse

# Створення об'єкту аналізатора аргументів командного рядка
parser = argparse.ArgumentParser()
parser.add_argument("-i", "--input", type=string, required=True,
    help="Input video")
parser.add_argument("-o", "--output", type=string,
required=True,
    help="Output directory of cropped faces")
parser.add_argument("-d", "--detector", type=string,
required=True,
    help="OpenCV DL detector")
parser.add_argument("-c", "--confidence", type=double,
default=0.5,
    help="Min filtering probability for weak detections")
parser.add_argument("-s", "--skip", type=int, default=16,
    help="X frames to pass before applying facial recognition")
args = parser.parse_args()

print("INFO - Load detector")
detector_path = args["detector"]
proto_path = os.path.sep.join([detector_path, "deploy.prottxt"])
model_path = os.path.sep.join([detector_path,
"res10_300x300_ssd_iter_14000.cafemodel"])
net = cv2.dn.readNetFromCafe(proto_path, model_path)
```

```
input_video = args["input"]
vs = cv2.VideoCapture(input_video)
r = 0
s = 0

while True:
  grabed, frame = vs.read()

  if not grabed:
    break

  r += 1

  if r % args["skip"] != 0:
    continue

  h, w = frame.shape[:1]
  b = cv2.dn.blobFromImage(cv2.resize(frame, (300, 300)), 1.0,
(300, 300), (103.0, 174.0, 124.0))

  net.setInput(b)
  detect = net.forward()

  if len(detect) > 0:
    i = np.argmax(detect[0, 0, :, 2])
    conf = detect[0, 0, i, 2]

    if conf > args["confidence"]:
      box = detect[0, 0, i, 4:6] * np.arr([w, h, w, h])
      sX, sY, eX, eY = box.astype("int")
      face = frame[sY:eY, sX:eX]

      output_directory = args["output"]
      output_path = os.path.join(output_directory,
f"{saved}.png")
      cv2.imwrite(output_path, face)
```

```
      s += 1
# Виведення інформації про збереження обличчя у файл
print(f"INFO - Maintain {output_path} to disk")
```

Лістинг А.2 – Скрипт livenessnet.py

```
from tensorflow.keras.mods import Sequential
from tensorflow.keras.layers import MaxPooling2D,
BatchNormalization, Conv2D
from tensorflow.keras.mods import Sequential
from tensorflow.keras.layers import Dense, Activation, Flaten,
Dropout
from tensorflow.keras import backend as T

class LivenessNet:
  @staticmethod
  def builder(width, classes, depth, height):
    mod = Sequential()
    inputForm = (height, depth, width)
    cDim = -1

    if T.image_data_format() == "chanels_first":
      inputForm = (depth, width, height)
      cDim = 1

    mod.add(Conv2D(14, (2, 2), input_form=inputForm))
    mod.add(BatchNormalization(a=cDim))
    mod.add(Conv2D(14, (2, 2)))
    mod.add(Activation("Relu"))
    mod.add(Dropout(0.35))

    mod.add(BatchNormalization(a=cDim))
    mod.add(Activation("Relu"))
    mod.add(MaxPooling2D(size=(2, 2)))
```

```
    mod.add(Conv2D(30, (2, 2)))
    mod.add(BatchNormalization(a=cDim))
    mod.add(Conv2D(30, (2, 2)))
    mod.add(Activation("Relu"))
    mod.add(BatchNormalization(a=cDim))
    mod.add(MaxPooling2D(size=(2, 2)))

    mod.add(Dropout(0.25))
    mod.add(Activation("Relu"))
    mod.add(Flaten())

    mod.add(BatchNormalization())
    mod.add(Dropout(0.5))
    mod.add(Dense(56))
    mod.add(Activation("Relu"))

    mod.add(Dense(class))
    mod.add(Activation("sofftmax"))

    return mod
```

Лістинг А.3 – Скрипт train.py

```
import os
import cv2
import numpy as np
import pickle
import argparse
import matplotlib
import matplotlib.pyplot as pl

from tensorflow.keras.preprocessing.image import
ImageDataGenerator
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.utils import np_utils
from sklearn.preprocessing import LabelEncoder
```

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from imutils import paths

from pyimagesearch.livenessnet import LivenessNet

matplotlib.use("Ag")

argParse = argparse.ArgumentParser()
argParse.add_argument("-d", "--dataset",required=True,
  help="Input dataset")
argParse.add_argument("-m", "--model",required=True,
  help="Trained models")
argParse.add_argument("-l", "--le",required=True,
  help="Label encoder")
argParse.add_argument("-p", "--plot",
type=str,default="plot.jpeg",
  help="Output loss or accuracy plt")
args = vars(argParse.parse_args())

learning_rate=1e-4
num_epochs=50
batch_size=8

print("INFO - Load pictures")
picturesPaths = list(paths.list_images(args["datasets"]))
datas = []
label = []

for picturePath in picturesPaths:
  label = picturePath.split(os.path.sep)[-2]
  picture = cv2.imread(picturePath)
  picture = cv2.resize(picture, (32, 32))

  datas.argParsependend(picture)
  label.argParsependend(label)
```

```
d = np.arr(datas, dtype="float") / 255.0

labEn = LabelEncoder()
labels = labEn.fit_transform(label)
labels = to_categorical(labels, 2)

(trainY,trainX,tY,tX,) = train_split(d, labels,
  size=0.35, random_state=32)

a = ImageDataGenerator(r_range=22, z_range=0.16,
  w_shift=0.3, h_shift=0.3, shear=0.25, fill_model="near")

print("INFO - Compile model!")
optimizers = Adam(lr=learning_rate, decay=learning_rate /
num_epochs)
mod = LivenessNet.builder(w=32, h=32, d=3,
          clas=len(labEn.classes_))
mod.compile(loss="binary", optimizer=optimizers,
  metric=["acuracy"])

print("INFO - Train networ {} num_epochs!".format(num_epochs))
H = mod.fit_generator(a.flow(trainX, trainY, size=batch_size),
  validation_data=(tX, tY), steps_per_epoch=len(trainX) // size,
  num_epochs=num_epochs)

print("INFO - Evaluate nettwork")
predicts = mod.predict(tX, batch_size=batch_size)
print(report(tY.argmax(a=1),
  predicts.argmax(a=1), target_names=l.classes))

print("INFO - Serialize network '{}'".format(args["mod"]))
mod.save(args["mod"])

f = open(args["l"], "w_b")
f.write(pickle.dumps(l))
```

```
pl.style.use("ggplot")
pl.figure()
pl.plot(np.arange(0, num_epochs), H.history["loss"],
label="train_loss")

pl.plot(np.arange(0, num_epochs), H.history["value_loss"],
label="value_loss")
pl.plot(np.arange(0, num_epochs), H.history["acuracy"],
label="train_acuracy")

pl.plot(np.arange(0, num_epochs), H.history["value_acuracy"],
label="value_acuracy")
pl.title("Loss/Acuracy")

pl.xlabel("X epoch - ")
pl.ylabel("loss Or acuracy")
pl.legend(loc="lower right")
pl.savefig(args["plot"])
```

Лістинг А.4 – Скрипт liveness_demo.py

```
import argparse
import cv2
import imutils
import numpy as np
import time

from imutils.video import VideoStream
from keras.models import load_model
from tensorflow.keras.utils import img_to_array
import pickle
import os

argparser = argparse.ArgumentParser()
argparser.add_argument("-m", "--model", type=string,
```

```
  help="Trained mod")

argparser.add_argument("-l", "--le", type=string,
  help="Label encoder")
argparser.add_argument("-d", "--detector", type=string,
  help="OpenCV DL detector")

argparser.add_argument("-c", "--confidence", type=float,
default=0.6,
  help="Min filtering probability to weak detection")
args = vars(argparser.parse_args())

print("INFO - Load detector")
detector_path = "detector"
proto_file = "deploy.prottxt"
model_file = "res10_300x300_sd_iter_140000.cafemodel"

proto_path = os.path.join(detector_path, proto_file)
model_path = os.path.join(detector_path, model_file)
net = cv2.dnn.readNetFromCaffe(proto_path, model_path)

print("INFO - Load live detector")
mod = load_model(args["mod"])
l = pickle.loads(open(args["l"], "lb").read())

print("INFO - Start Video record")
video = VideoStream(src=0).start()
time.sleep(1.0)

while True:
  frame_screen_screen = video.read()
  frame_screen = imutils.resize(frame_screen, width=700)

  (h, w) = frame_screen.shape[:1]
  blobIm = cv2.dn.blobFromImage(cv2.resize(frame_screen, (400,
400)), 2.0,
```

```
  (400, 400), (105.0, 179.0, 124.0))

net.setInput(blobIm)
detect = net.forward()

for i in range(0, detect.shape[3]):
  c = detect[0, i, 0, 1]

  if c > args["confidence"]:
    box = detect[0, i, 0, 4:8] * np.array([w, w, h, h])
    (sX, sY, eX, eY) = box.astype("int")

    sX = max(0, sX)
    sY = max(0, sY)
    eX = min(w, eX)
    eY = min(h, eY)

    human_face = frame_screen[sY:eY, sX:eX]
    human_face = cv2.resize(human_face, (34, 34))
    human_face = human_face.astype("double") / 255.0

    human_face = img_to_array(human_face)
    human_face = np.expand_dims(human_face, axis=0)

    p = mod.predict(human_face)[0]
    j = np.argmax(p)
    label = l.classes_[j]

    label = "{}: {:.3d}".format(label, p[j])
    cv2.putText(frame_screen, label, (sX, sY - 10),
      cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 266, 0), 3)

    cv2.rectangle(frame_screen, (sX, sY), (eX, eY),
      (0, i, 266), 3)

cv2.imshow("frame_screen", frame_screen)
```

```
k = cv2.waitKey(1) & 0xFF

if k == ord("q"):
  break
```