Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

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

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

РОЗРОБКА ТА ДОСЛІДЖЕННЯ МЕТОДУ ДЛЯ МОНІТОРИНГУ ДІЙ УЧАСНИКІВ ОНЛАЙН-КОНФЕРЕНЦІЙ НА ОСНОВІ АНАЛІЗУ ВІДЕОПОТІКУ
(тема)

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

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Яковлева О.В.
(посада, прізвище, ініціали)

Допускається до захисту

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

2022 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

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

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Ардасову Вадиму Андрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка та дослідження методу для моніторингу дій учасників онлайн-конференцій на основі аналізу відеопотіку
затверджена наказом по університету від 9 листопада 2022 року № 1469Ст
2. Термін подання студентом роботи до екзаменаційної комісії 25 листопада 2022 р.
3. Вихідні дані до роботи математичні моделі вирівняння обличчя за позицією очей, метод детектування обличчя Хаара, штучні нейроні мережі MTCNN, MXNet, SSD MobileNet v2, нейроні мережі верифікації обличчя VGG-Face, FaceNet, OpenFace DeepFace, ArcFace, метод отримання характерних точок кісті руки та обличчя, нейрона мережа детектування емоцій, Angular, .Net, AspNet Core, SignalR, RabbitMQ, Redis, OpenCV, Anaconda.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд методів детектування обличчя.
2. Огляд методів верифікації обличчя.
3. Огляд методів детектування кісті руки та характерних точок кісті.
4. Огляд методів аналізу емоцій за обличчям.
5. Розробка та реалізація алгоритму моніторингу дій учасників відео конференції.
6. Доступ до відеопотіку учасників відеоконференції Zoom.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність поставленої проблеми, постановка задачі, використані технології, етапи роботи, алгоритм роботи моніторингу дій учасників відеоконференції, детектування обличчя, верифікація обличчя, детектування кісті руки, розпізнання емоцій, доступ до відеопотіку відеоконференції, отримання характерних точок, результат алгоритму, статистичні дані та їх аналіз, висновки.

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

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

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	09.11.2022	
2	Аналіз завдання, підбір літератури	10.11.22-12.11.22	
3	Аналіз літератури з досліджуваної проблеми	12.11.22-13.11.22	
4	Аналіз технічних засобів, бібліотек та сучасний стан питання	13.11.22	
5	Розробка методу для моніторингу дій учасників онлайн-конференцій на основі аналізу відеопотіку	14.11.22	
6	Програмна реалізація	15.11.22-18.11.22	
7	Оформлення пояснювальної записки	18.11.22-19.11.22	
8	Перевірка на плагіат	25.11.2022	
9	Рецензування	25.11.2022	
10	Підготовка презентації та доповіді	27.11.2022	
11	Занесення роботи в електронний архів	30.11.2022	
12	Попередній захист кваліфікаційної роботи	05.12.2022	

Дата видачі завдання 9 листопада 2022 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Яковлева О.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 59 с., 15 табл., 29 рис., 42 джерела.

ДЕТЕКЦІЯ ОБЛИЧЧЯ, ВЕРИФІКАЦІЯ, РОЗПІЗНАННЯ ЕМОЦІЙ, ДЕТЕКТУВАННЯ ЖЕСТІВ РУКИ, FACENET, VGG, MTCNN, HAAR, SSD, MXNET, OPENFACE, ARC FACE, DEEPFACE, MEDIAPIPE.

Об’єктом дослідження є відеопотік учасників сесії відео конференції сервісу онлайн-конференцій.

Метою дослідження є розробка методу для моніторингу дій учасників онлайн-конференцій на основі аналізу відеопотіку.

Використано методи глибоких нейронних мереж. Проведено дослідження методів детектування обличчя, верифікації, емоцій на основі методу характерних точок, кісті руки, та аналіз отриманих даних. Досліджено метод моніторингу дій учасників сесії онлайн відео конференції.

У результаті дослідження розроблений програмний застосунок SmartGuard.

FACE DETECTION, VERYFICATION, EMOTION DETECTION, HANG GASTURE DETECTION, FACENET, VGG, MTCNN, HAAR, SSD, MXNET, OPENFACE, ARC FACE, DEEPFACE, MEDIAPIPE.

The object of the research is participants video stream of online video conference.

The aim of the research is implementation and researching a method for monitoring the actions of participants in online conferences based on video stream analysis.

The methods of deep neural networks are used. The research was conducted of methods of face detection, verification, emotions based on the method of characteristic points, hand, and analysis of the data obtained. The method of monitoring the actions of participants in an online video conference session is researched.

As a result of the research, the SmartGuard software application was developed.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ПЗ – програмне забезпечення

ШНМ – штучна нейронна мережа

JWT – Json Web Token (вебтокен)

CV – Computer Vision (комп’ютерний зір)

JS – JavaScript

SSD – Single Shot Detector (детектор одиночних пострілів)

МН – машинне навчання

SDK – Software Development Kit (набір для розробки)

ШІ – штучний інтелект

CPU – Central Processing Unit (центральний процесор)

GPU – Graphics Processing Unit (графічний процесор)

ВСТУП

З розвитком технологій для багатьох сфер людського буття відкриваються нові можливості, те що здавалося неможливим учора, становиться повсякденним сьогодні. COVID-19 вплинув на всі сфери людського буття, починаючи від бізнесу закінчуючи процесами навчання в школах та закладах вищої освіти. Сервіси онлайн відеоконференцій стали також невідємлемою частиною життя: наради, бізнес зустрічі, конференції, лекції, проведення екзаменів та іспитів.

З новими технологіями та часом приходить нові правили, яких потрібно дотримуватися, так від студентів вимагають під час онлайн іспитів та усієї сесії вмикати камери, щоб знизити ризики шахрайства. Але ще не існує сервісів, які можуть підтримувати правила проведення сесії автоматично, використовуючи сучасні технології детектування, верифікації та збору статистичних даних в режимі реального часу.

Існують аналоги, котрі допомагають комісії стежити за правилами проведення на прикладі Duolingo English Test, але найголовніший мінус цього сервісу, що це постобробка запису сесії та підтримуються лише один учасник.

Актуальність дослідження полягає у розробці методу для моніторингу дій учасників відео конференцій за допомогою аналізу відеопотіку, кожного учасника в режимі реального часу, використовуючи сучасні підходи та моделі штучних нейронних мереж для автоматизації процесу підтримки правил проведення сесії. Це допоможе підтримувати якість проведення онлайн іспитів, зменшить відсоток шахрайства та полегшить роботу комісії по відстеженню підозрілих моментів.

В ході роботи буде розроблений метод моніторингу та реалізований програмний застосунок для доведення ідеї та методу, використовуючи сучасний підхід для побудови ПЗ.

1 СУЧАСНИЙ СТАН ПИТАННЯ МОНІТОРИНГУ ДІЙ УЧАСНИКІВ ОНЛАЙН-КОНФЕРЕНЦІЙ НА ОСНОВІ АНАЛІЗУ ВІДЕОПОТІКУ

1.1 Огляд сервісів для онлайн-конференцій в порівняльному аспекті

Відеоконференції стали невід'ємним повсякденним бізнес-інструментом для IT сектору, школярів, студентів вищих навчальних закладів тощо і за останні роки технології не стоять на одному місці та постійно розвиваються, незважаючи на обмеження, пов'язані з COVID-19 та інші, які заперечують повсякденному ходу життя. Лекції, екзамени, лабораторні роботи, наради та робота – все, що вимагає комунікації, проводяться за допомогою сервісів відеоконференцій.

Зі зростаючим попитом з'являється багато пропозиції в обличчі найпопулярніших сервісів для проведення онлайн-конференцій. Розглянемо 3 найпопулярніших сервісів для проведення онлайн відеоконференцій Zoom, Google Meet, та Microsoft Teams.

Google Meets – застосунок для відеоконференцій в Google Workspace, раніше G Suite, був спеціально розроблений з урахуванням потреб бізнесу. Це простий у використанні інтерфейс, який може обслуговувати до 250 осіб під час онлайн-наради в режимі відеоконференції, залежно від підписки на Google Workspace. Простий у використанні інтерфейс, повністю інтегрований з іншими застосунками Google Workspace, такими як Google Calendar, дозволяє користувачам швидко створювати наради, приєднуватися до них і виходити з них, просто натиснувши на гіперпосилання.

Zoom – відмінний варіант для відео конференцій, зустрічей та віддаленої роботи. Він надає всі необхідні функці, простий у використанні, а якість відео та аудіо висока. Можливо легко організувати онлайн-зустрічі та запросити членів своєї команди, просто поділившись гіперпосиланням, та містить

програмні застосунки, доступні для всіх пристроїв, у тому числі мобільних телефонів.

Zoom пропонує безкоштовний тарифний план, який включає дзвінки для не більш ніж 100 учасників і необмежену кількість зустрічей один на один. Цього більш ніж достатньо для більшості типових задач, наприклад, проведення онлайн занять. Недоліком безкоштовного плану є те, що тривалість онлайн-конференцій обмежена 40 хвилинами, та не надає безкоштовний виділений віртуальний телефонний номер для зв'язку. Однак кожна Zoom конференція отримує унікальний телефонний код, який учасники можуть використовувати, якщо у них немає програми або надійного стабільного доступу до Інтернету або електромережі.

Microsoft Teams – це рішення для відеоконференцій з декількома планами обслуговування та більш спрямований на бізнес та організації. Teams є частиною Microsoft Office 365, тому багато Microsoft сервісів інтегровано з Teams та навпаки.

При роботі з великою групою простий інтерфейс Microsoft Teams спрощує роботу. За допомогою декількох кліків можна швидко створити канал і організувати розмови в окремі потоки. Оскільки всі ці функції містяться в одному інструменті, можливо легко переключитися з чату на відео або телефонну розмову.

З боку можливостей інтеграції платформи відеоконференцій значними лідерами в цьому питанні є Zoom та Microsoft Teams, але виграє Zoom з великою кількістю різноманітних API та SDKs, які він надає розробникам: Zoom API, Phone API, Contact Center API, Video SDK API, QSS Api, Chat API, Meeting SDK та інші.

За результатами порівняння Zoom є значним переможцем в сфері сервісів відеоконференцій через свою гнучкість та функціонал. Крім того Zoom має свій власний Market Place, де розробники можуть інтегрувати свої програмні рішення безпосередньо в Zoom застосунок або навпаки Zoom в бізнес рішення.

Порівняння сервісів відеоконференцій наведено в таблиці 1.1.

Таблиця 1.1 – Порівняння сервісів відеоконференцій

Функціонал	**Zoom**	**Teams**	**G Meet**
Синхронізація календаря	+	-	-
Шифрування та парольний захист	+	+	+
Віртуальні зали очікування	+	+	+
Відео та аудіо високої чіткості (HD)	+	-	-
Показ екрана	+	+	+
Запис та стенограми	+	+	+
Живий чат	+	+	+
Інструменти залучення («підняття руки», реакції, опитування)	+	+	+
Миттєвий обмін файлами та редагування в режимі реального часу	-	+	-
Інтеграція з бізнес-застосунками	-	+	+
API/SDK	+	+	-

Для створення власної інтеграції розробник повинене створити окремий акаунт на платформі Zoom для розробників та зареєструвати свій продукт на ній.

Zoom пропонує такі варіанти типів застосунків:

– Zoom Apps для застосунків, які додаються та використовуються в Zoom Client;

– JWT для застосунків, що підтримують міжсерверну інтеграцію з сервісами Zoom без необхідності авторизації користувача;

– OAuth для застосунків, які отримують доступ до аутентифікованих даних користувача для використання з застосунками третіх сторін (thirdparty);

– Chat Apps для застосунків для чату в Zoom, який встановлюється в Zoom Client та взаємодіє з користувачами через чат;

– Meeting SDK для створення мобільних, десктопних, прогресивних веб-застосунків та гібридних, які інтегрують функції Zoom Client;

– Webhook Only для застосунків, котрі хочуть отримувати сповіщення на основі подій для облікового запису Zoom, таких як зустрічі, вебінари, хмарні записи тощо;

– Server-to-Server OAuth для створення застосунків, якщо планується, що застосунок буде використовуватися тільки одним користувачем облікового запису Zoom. Цей застосунок найкраще використовувати для створення внутрішнього інструменту або з'єднувача для кращого управління обліковим записом Zoom;

– Video SDK для власних застосунків на базі Zoom функціоналу, що забезпечує передачу відео, аудіо, спільний доступ до екрану, чат, потоки даних та багато іншого в якості сервісу.

1.2 Сучасний стан питання розпізнавання обличчя

Розпізнавання обличчя – це метод ідентифікації або підтвердження особи людини за її обличчям. Системи розпізнавання обличчя можуть використовуватися для ідентифікації людей на фотографіях, відео або в режимі реального часу [1].

Дослідження питання розпізнавання обличчя ведуться вже майже 50 років. Розпізнавання обличчя є одним з напрямків досліджень в області розпізнавання образів та комп'ютерного зору завдяки своїм численним практичних застосувань в області біометрії, інформаційної безпеки, контролю доступу, правоохоронних органів, смарт-карт і системи спостереження.

В останні роки біометричні методи стали найбільш перспективним варіантом розпізнавання осіб, оскільки замість того, щоб засвідчувати осіб та надавати їм доступ до фізичних та віртуальних доменів на основі паролів, PIN-кодів, смарт-карт, пластикових карток, токенів, ключів тощо, ці методи досліджують фізіологічні або поведінкові характеристики особи з метою визначення або встановлення її особистості. Паролі та PIN-коди важко запам’ятати, їх можна вкрасти або вгадати; картки, жетони, ключі тощо можна загубити, забути або продублювати. Однак біологічні ознаки людини неможливо загубити, забути, викрасти або підробити [2].

Для того, щоб розробити систему розпізнавання обличчя необхідно враховувати декілька факторів:

– загальна швидкість роботи системи від детектування до розпізнавання повинна бути задовільною;

– точність результату повинна бути високою;

– система повинна легко оновлюватися та масштабуватися, тобто легко збільшувати кількість об’єктів, які можуть бути оброблені, проаналізовані та розпізнані.

Розпізнавання обличчя є складною і водночас цікавою проблемою, що привертає увагу дослідників з різних галузей: психології, розпізнавання образів, нейронних мереж, комп’ютерного зору та комп’ютерної графіки. Для розпізнавання обличчя використовуються наступні методи:

– комплексні методи узгодження;

– ознакові (структурні) методи або Feature-based;

– гібридні методі.

При комплексному підході в якості вхідних даних для системи розпізнавання обличчя враховується вся область обличчя. Одними з найкращих прикладів комплексних методів є Eigenfaces [3], аналіз головних компонент, лінійний дискримінантний аналіз та аналіз незалежних компонент тощо.

Методи на основі ознак виділяють локальні ознаки, такі як очі, ніс і рот, а їх місцезнаходження та локальна статистика вводяться в структурний класифікатор. Великим викликом для методів вилучення ознак є відновлення ознак, коли система намагається отримати ознаки, які є невидимими через варіативність зображення, наприклад, положення голови, коли зіставляється фронтальне зображення із зображенням профілю. Розрізняють три різні методи вилучення ознак:

– загальні методи, засновані на краях, лініях та кривих;

– методи на основі шаблонів ознак;

– структурні методи зіставлення, які враховують геометричні обмеження на особливості обличчя.

Гібридні системи розпізнавання обличчя використовують комбінацію як комплексних методів узгоджень, так і методи виділення ознак. Як правило, в гібридних методах використовуються 3D-зображення. Зображення обличчя людини знімається в 3D, що дозволяє системі відзначити особливості очних ямок або форми підборіддя або чола. Навіть обличчя в профіль може бути використано в гібридних системах, оскільки використовує глибину зображення, що дає достатньо інформації для побудови повного обличчя.

Кілька компаній змагаються за першість у розробці біометричних технологій, включаючи Google та Apple. Теоретичні відкриття у сфері штучного інтелекту, розпізнавання зображень та аналізу обличчя регулярно публікуються всіма основними гравцями: Amazon, Microsoft і Facebook.

У 2014 році компанія Facebook анонсувала свою програму DeepFace [4], яка може розпізнати, чи належать два сфотографованих обличчя одній людині, з точністю до 97,25%. При проходженні такого ж тесту люди відповідають правильно у 97,53% випадків, або лише на 0,28% краще, ніж програма Facebook.

У червні 2015 року Google покращив свої показники за допомогою моделі FaceNet [5]. FaceNet встановив новий еталон точності на широко використовуваному наборі даних Labeled Faces in the Wild (LFW) [6],

досягнувши результату 99,63%. Ця технологія включена в Google Photos і використовується для сортування фотографій і автоматичного позначення їх тегами на основі розпізнаних людей.

Дослідження, проведене дослідниками показало, що інструменти Microsoft, IBM (Face++) [7] мають високий рівень помилок при ідентифікації темношкірих жінок у порівнянні зі світлошкірими чоловіками, але за останній час Microsoft вдосконалила свою технологію упередженого розпізнавання осіб.

У сучасних технологіях розпізнавання обличчя є 4 загальних етапи. Вони включають детектування, вирівнювання, представлення та верифікація або розпізнавання (рис. 1.1).

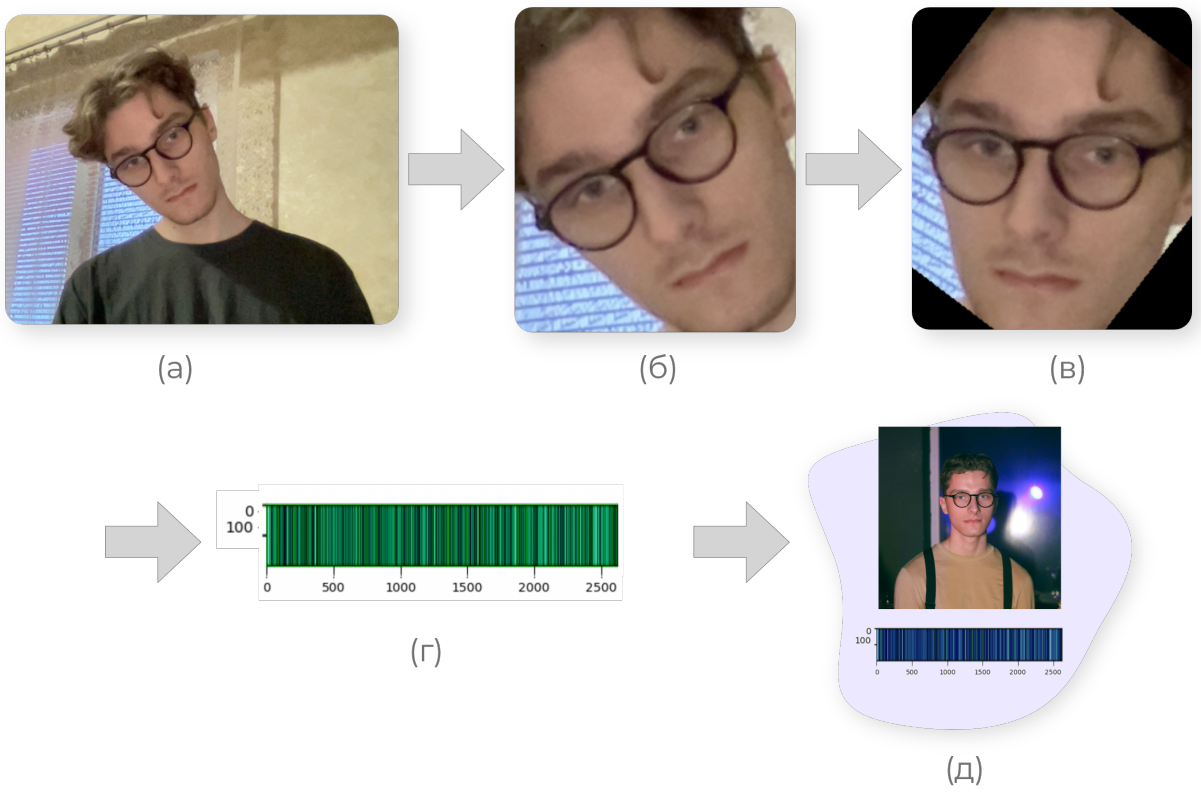


Рисунок 1.1 – Етапи розпізнавання обличчя:
(а) вихідне зображення; (б) детектування обличчя;
(в) вирівнювання; (г) представлення; (д) верифікація

1.3 Бібліотеки Computer Vision для аналізу обличчя та дій людини

Існує багато задач, які вивчають Computer Vision, багато з них вже вирішенні та досить оптимізовані, щоб працювати в режимі реального часу, та сформовані в відкриті opensource бібліотеки, які доступні для вирішення бізнес задач та більш специфічних проблем:

– детектування обличчя на зображенні;

– розпізнавання обличчя;

– створення Face Mesh або лицьової сітки;

– детектування рук, пози, скелета тіла;

– сегментація, відділення волосся, усього тіла або його частин.

Одною з найпопулярніших бібліотек є OpenCV [8] – це бібліотека з відкритим вихідним кодом, яка була розроблена компанією Intel ще, у 2000 році. Здебільшого вона використовується в задачах комп'ютерного зору, таких як виявлення об'єктів, виявлення обличчя, розпізнавання обличчя, сегментація зображень тощо, але також містить багато корисних функцій, які можуть знадобитися в машинному навчанні.

DeepFace – бібліотека, що спрямовано здебільш на обробку обличчя: детектування, розпізнавання, аналіз обличчя, вік, стать, емоції та підтримує різні методи, такі як FaceNet та InsightFace [9]. Вона також надає REST API, але підтримує лише для методи верифікації.

FaceNet – бібліотека на мові програмування Python з відкритим вихідним кодом. Точність цього методу досить висока – 99,65% на датасеті LFW, що є гарним, але не найвищим показником. Недоліками бібліотеки є те що репозиторій більше не підтримується та не отримує оновлень.

Keras – бібліотека з відкритим вихідним кодом на основі Python [10], яка виступає в якості інтерфейсу для платформи машинного навчання TensorFlow [11]. Вона особливо підходить для задач, моделей та алгоритмів пов'язаних з машинним навчанням оскільки дозволяє швидко побудувати нейромережеву модель, забезпечуючи при цьому бекенд-підтримку.

PyTorchCV – бібліотека на основі PyTorch для задач комп'ютерного зору. Бібліотека містить в собі набір моделей класифікації, сегментації, виявлення та оцінки різних типів зображень. У фреймворку реалізовано низку моделей, серед яких AlexNet, ResNet, ResNeXt, PyramidNet, SparseNet, DRN-C/DRN-D та інші.

MediaPipe – це бібліотека для побудови конвеєрів машинного навчання для обробки відео та аудіо даних [12]. Бібліотека є кросплатформенною тому може працювати на багатьох видах пристроїв, використовуючи API на багатьох мовах специфічних для платформи: Android/Java, iOS, C++, Python і навіть JS. Бібліотека містить багато вже готових рішень для задач комп'ютерного зору, заснованих на конкретній попередньо навченій моделі TensorFlow або TFLite (рис. 1.2).

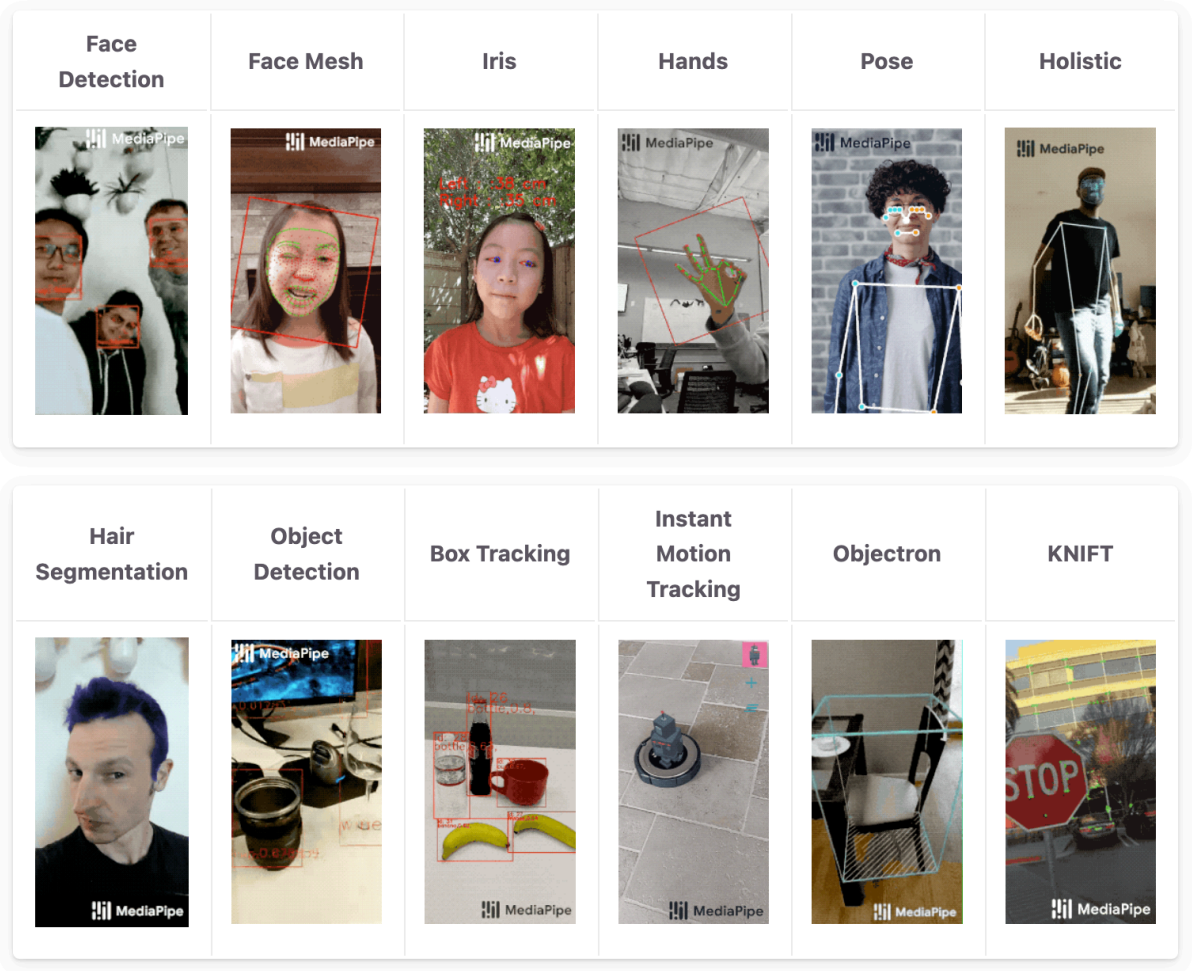


Рисунок 1.2 – Демонстрація функціоналу MediaPipe

1.4 Постановка задачі дослідження

Актуальність дослідження полягає у розробці методу для моніторингу дій учасників відео конференцій за допомогою аналізу відеопотіку, кожного учасника в режимі реального часу, використовуючи сучасні підходи та моделі штучних нейронних мереж для автоматизації процесу підтримки правил проведення сесії.

Об’єктом дослідження є відеопотік учасників сесії конференції сервісу онлайн-конференцій.

Метою дослідження є розробка методу для моніторингу дій учасників онлайн-конференцій на основі аналізу відеопотіку.

Для досягнення мети необхідно вирішити такі завдання:

– ознайомлення та порівняння сервісів онлайн-конференцій;

– впровадження та інтеграція сервісів онлайн-конференцій для програмного застосунку;

– доступ до відеопотіку кожного учасника конференції;

– ознайомлення, вивчення та впровадження нейронних мереж для задачі детектування обличчя;

– ознайомлення, вивчення та впровадження нейронних мереж для задачі верифікації обличчя;

– ознайомлення, вивчення та впровадження нейронних мереж для задачі класифікації емоцій, віку, статі та раси;

– дослідження та розробка методу моніторингу дій учасників за відеопотіком конференції;

– розробка, проєктування програмного застосунку для реалізації методу моніторингу дій учасників;

– проєктування бази даних для зберігання результатів конференції;

– розробка методу аналізу отриманих даних записаних під час сесії відео конференції та їх відображення для подальшого аналізу.

2 РОЗРОБКА МЕТОДУ МОНІТОРІНГУ ДІЙ УЧАСНИКІВ ОНЛАЙН-КОНФЕРЕНЦІЙ ZOOM

2.1 Верифікація облич учасників

Сучасні алгоритми верифікації або розпізнавання обличчя це не лише одна модель, а ланцюг, конвеєр або пайплайн (pipeline) моделей та алгоритмів. В загальному випадку існують чотири етапи сучасного пайплану розпізнавання обличчя – це детектування, вирівнювання, виявлення характерних ознак та класифікація або верифікація. Етапи детектування та вирівнювання мають велике значення через варіативність фото або відеопотіку (вхідних даних), наприклад обличчя може знаходитися в профіль або анфас і ці нюанси мають вирішальне значення для роботи, пов’язаної з розпізнаванням обличчя.

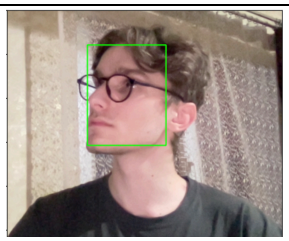
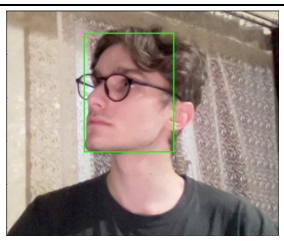
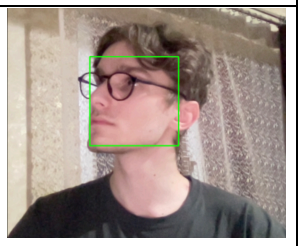
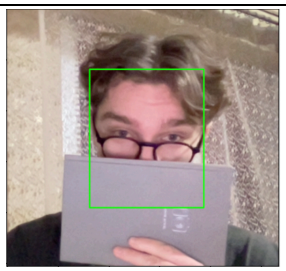
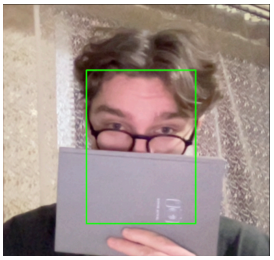
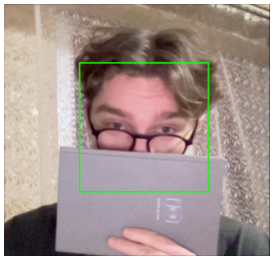
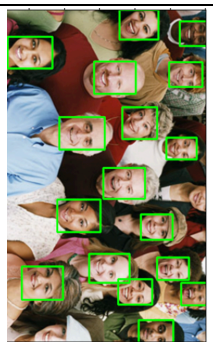
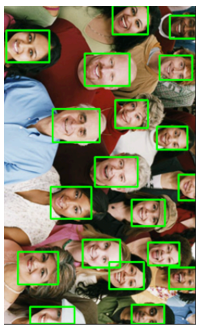
Існує багато методів детектування обличчя, серед яких MTCNN, каскадний метод Хаара (Haar) [13] реалізовані в бібліотеках OpenCV [14], Dlib, та однокадрове багатоблочне виявлення (SSD) реалізовані в опенсорс бібліотеці OpenCV. Виявлення об’єктів з максимальною різницею та гістограма орієнтованих градієнтів (HOG) реалізовані в Dlib. І останнє, але не менш важливе, MTCNN [15] є популярним варіантом у спільноті з відкритим вихідним кодом. Метод Haar і HoG є застарілими методами, в той час, як SSD [16], MMOD і MTCNN є сучасними методологіями, заснованими на глибокому навчанні, крім того, SSD є найшвидшим.

За результатами порівняння методів детектування обличчя Haar показав себе, як швидкий алгоритм, але не стійкий до положення обличчя, MTCNN в свою чергу показує гарні результати за часом та задовільною точністю оцінки. MXNet [17] як і очікувалось, показав вражаючу точність, але надто повільний. SSD на базі MobileNet v2 [18] показав вражаючу швидкість детектування,

задовільну точність, але не підходить до зображень поганої якості або великим рівнем зашумленості чи високої плотністю об’єктів.

Результати порівняння наведені в таблиці 2.1.

Таблиця 2.1 – Порівняння методів детектування обличчя

Haar	**MTCNN**	**MXNet**	**SSD on MobileNet v2**
			
92,7 мс	789 мс	3,9 с	5,4 мс
			
46 мс	328 мс	3,1 с	3,2 мс
			
50 мс	445 мс	4,7 с	2,9 мс

Для задач нормалізації обличчя на зображенні можливо застосувати багато алгоритмів. Одні з них базуються на детектуванні очей, центру положення очей, та використовуючи формулу знаходження куту нахилу обличчя відносно осі ординат, вирівняти обличчя на фотографії [19]

$$\cos(A) = \frac{b^2+c^2-a^2}{2bc}, \tag{2.1}$$

$$l = \sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}, \tag{2.2}$$

де x_i, y_j – координати центру очей відносно зображення.

На рисунку 2.1 зображено алгоритм вирівнювання обличчя.

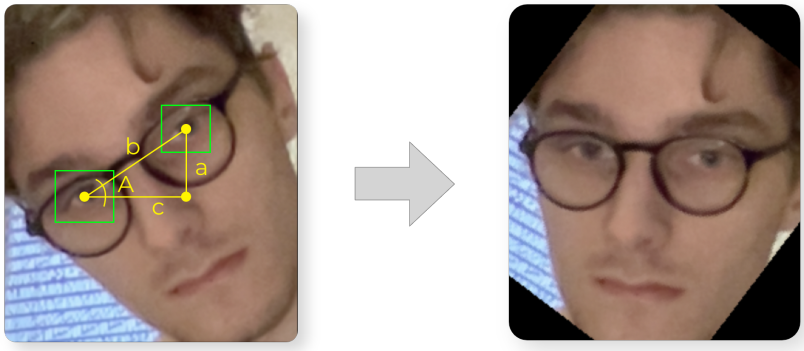


Рисунок 2.1 – Вирівняння обличчя шляхом детектування очей (Haar)

Виявлення характерних ознак або вектору обличчя сильно впливає на продуктивність системи розпізнавання обличчя, а також знаходиться в центрі уваги сучасних досліджень, пов'язаних з розпізнаванням. Одним з найточніших представників є штучна нейрона мережа (ШНМ) 16-шарова VGG-Face. ШНМ, яка була навчена на датасеті розміром більш ніж 2 мільйонах зображень.

Окрім виняткової продуктивності в тестах, VGG-Face [20] дуже проста в використанні завдяки тому, що можливо взяти вихідні дані шару fc-6 і виділити корисні ознаки зображення за допомогою VGG-Face, які потім будуть використанні на етапі класифікації. В результаті буде вектор ознак для кожного зображення.

На рисунку 2.2 схематично зображено архітектуру VGG-Face ШНМ та в таблиці 2.2.

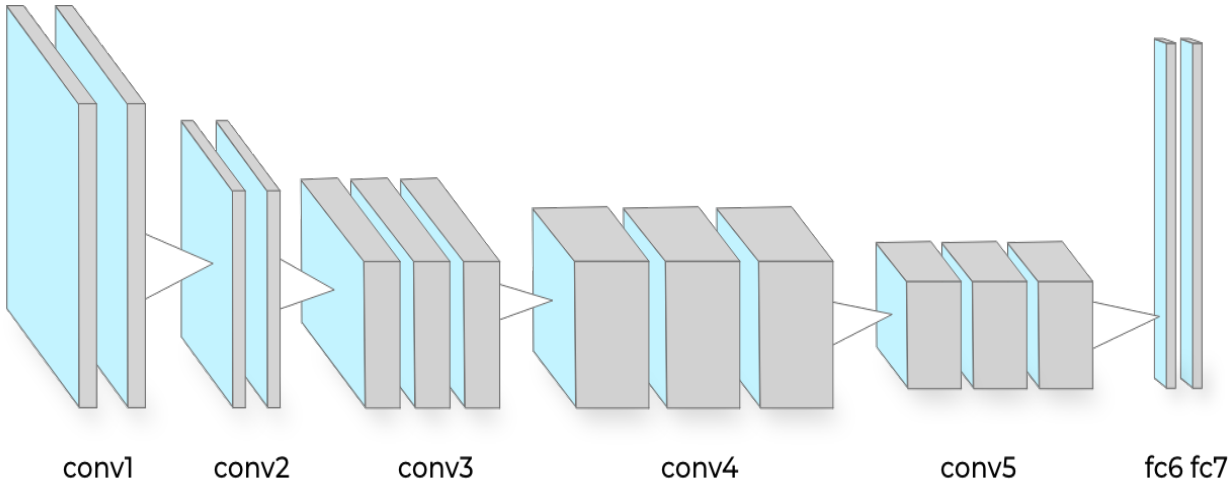


Рисунок 2.2 – Архітектура VGG-Face ШНМ

Таблиця 2.2 – Архітектура мережі VGG-16

Шар (тип)	**Вихідна форма**	**Параметри**
input_1 (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160

Продовження таблиці 2.2

Шар (тип)	**Вихідна форма**	**Параметри**
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
global_average_pooling2d_1	(None, 512)	0
dense_1 (Dense)	(None, 1024)	525312
dense_2 (Dense)	(None, 1024)	1049600
dense_3 (Dense)	(None, 512)	524800
dense_4 (Dense)	(None, 3)	1539

Після вилучення вектору ознак обличчя потрібно порівняти два вектори за метрикою відстані для вирішення задачі верифікації, тобто порівняти два зображення вихідне та надане. Якщо задача поставлена не як верифікація, а класифікація, або знаходження схожого вектору, то можливо використати методи класифікації такі як: Nearest-Neighbor [21] або SVM (Support Vector Machines) [22], але для задач верифікації, коли вже знаємо яке зображення є еталонним, це не потрібно.

В роботі використані 2 види метрик відстані:

– косинусна подібність

$$S_c(A, B) = \frac{A*B}{\|A\|\|B\|}; \tag{2.3}$$

– евклідова відстань

$$d(x, y) = \sqrt{(x - y)^2}. \quad (2.4)$$

Результати порівняння методів верифікації обличчя за моделями: VGG-Face, FaceNet, OpenFace [23], DeepFace, ArcFace [24] наведено в таблиці 2.3 Для детектування обличчя була використана ШНМ MTCNN, косинусна метрика відстані.

Зображення, котрі проходять верифікацію зображено на рисунку 2.3.

(а) (б) (в) (г) (д)

Рисунок 2.3 – Типи зображень:
(а) так; (б) так; (в) ні; (г) так; (д) еталоне зображення

Таблиця 2.3 – Порівняння методів верифікації обличчя

		VGG-Face	**FaceNet**	**OpenFace**	**DeepFace**	**ArcFace**
(а – д)	Verified	так	так	ні	ні	так
	Time	2,2 с	0,8 с	0,6 с	0,8 с	0,9 с
(б – д)	Verified	так	ні	ні	ні	так
	Time	1,4 с	0,7 с	0,5 с	0,6 с	0,8 с
(в – д)	Verified	так	ні	ні	ні	ні
	Time	1,3 с	0,7 с	0,6 с	0,6 с	0,8 с
(г – д)	Verified	так	так	ні	так	так
	Time	1,3 с	0,7 с	0,6 с	0,6 с	0,8 с

За результатами дослідження краще за всіх показала себе ArcFace на даному наборі зображень, що комбінує в собі високу точність оцінки та швидкодію. FaceNet також себе гарно показало, на зображеннях з гарною якістю. Гірше за всіх себе показала модель OpenFace за точністю оцінки.

2.2 Детектування піднятої руки

Здатність розпізнавати рух і форму рук може відігравати ключову роль у покращенні користувацького досвіду на різних технологічних платформах і в різних сферах.

Наприклад, вона може стати основою для управління жестами рук і розуміння жестів. Вона також може зробити можливим накладання цифрової інформації і матеріалів на реальний світ в доповненій реальності. Хоча для людей це відбувається без особливих зусиль, надійне сприйняття рук в реальному часі є надзвичайно складною проблемою комп’ютерного зору через те, що руки часто затуляють собою або одна одну і не мають висококонтрастних контурів [25].

MediaPipe Hands – це високоточне рішення для відстеження рук і пальців. Модель використовує машинне навчання для виведення 21 3D-орієнтирів руки лише з одного кадру. Модель забезпечує роботу в режимі реального часу і навіть масштабується до декількох рук, на відміну від існуючих сучасних систем, які здебільшого покладаються на потужність середи виконання (CPU, GPU).

Подібно до моделі виявлення обличчя модель детектора використовує підхід SSD, призначену для використання в режимі реального часу для виявлення початкового розташування рук.

Проблема виявлення рук є неймовірно складною, оскільки «легка» модель, так і повна модель повинні бути здатні розпізнавати закриті і полузачинені руки, працюючи в широкому діапазоні відносно кадру

зображення. Правильно ідентифікувати руки лише за їх візуальними ознаками досить складно, на відміну від обличчя, яке має висококонтрастну структуру, наприклад, в області рота та очей. Натомість точна локалізація рук стає можливою завдяки додаванню додаткового контексту, такого як аспекти руки, тулуба або особи.

Підхід використовує багато методів для вирішення вищезгаданих проблем. Першим в пайплайні є детектор долонь, а не детектор рук, оскільки набагато легше оцінити обмежувальні рамки жорстких об’єктів, таких як кулаки і долоні, ніж виявити руки зі пальцями. Метод не максимального придушення також добре працює в сценаріях двох рук, таких як рукостискання, оскільки долоні є меншими об’єктами. Кількість якорів зменшується в 3-5 разів, моделюючи долоні за допомогою квадратних обмежувальних рамок або якорів мовою машинного навчання. Для того, щоб забезпечити більшу обізнаність про контекст сцени, навіть для маленьких об’єктів, реалізовано екстрактор ознак кодера-декодера (подібно до підходу RetinaNet [26]), та для підтримки великої кількості якорів зменшено втрату фокусу під час навчання.

Після виявлення долоні по всьому зображенню, подальша модель орієнтирів руки використовує регресію, або пряме прогнозування координат, для досягнення точної локалізації ключових точок 3D координат суглобів кистей рук всередині виявлених областей руки. Модель набуває надійного внутрішнього представлення положення руки і не піддається впливу частково видимих рук.

Модель навчена на близько 30 тис. реальних фотографій, щоб отримати дані (береться значення Z з карти глибини зображення, якщо вона існує для відповідної координати). Додатково візуалізується високоякісна синтетична модель руки на різних фонах і прив’язується до відповідних 3D-координат, щоб краще охопити діапазон можливих поз рук і забезпечити додатковий нагляд за характером геометрії руки.

На рисунку 2.4 зображено модель руки на 21 характерну точку – результат, який надає алгоритм детектування руки на зображенні або відеопотіку. А на рисунку 2.5 зображено приклад використання детектування та накладання характерних точок руки на зображення.

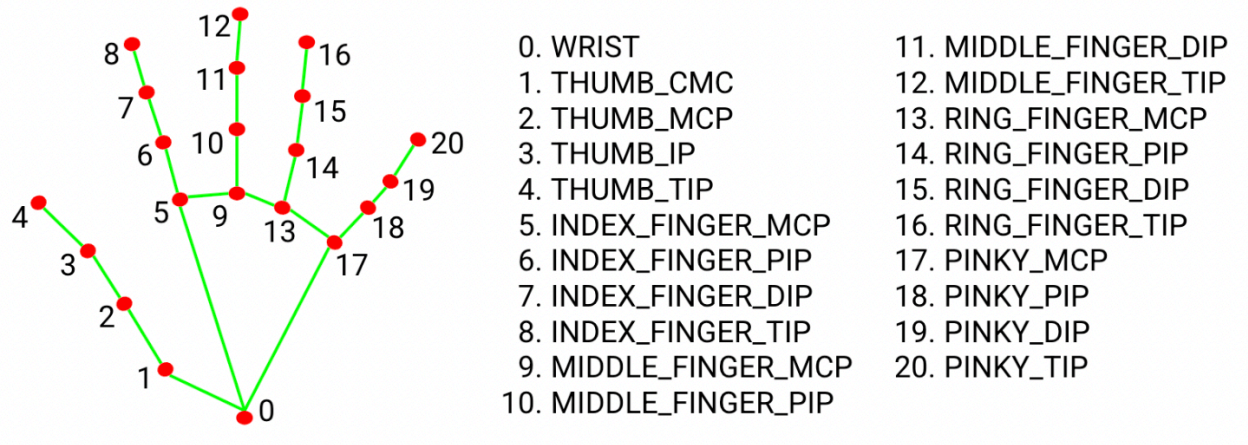


Рисунок 2.4 – Модель руки на 21 характерну точку

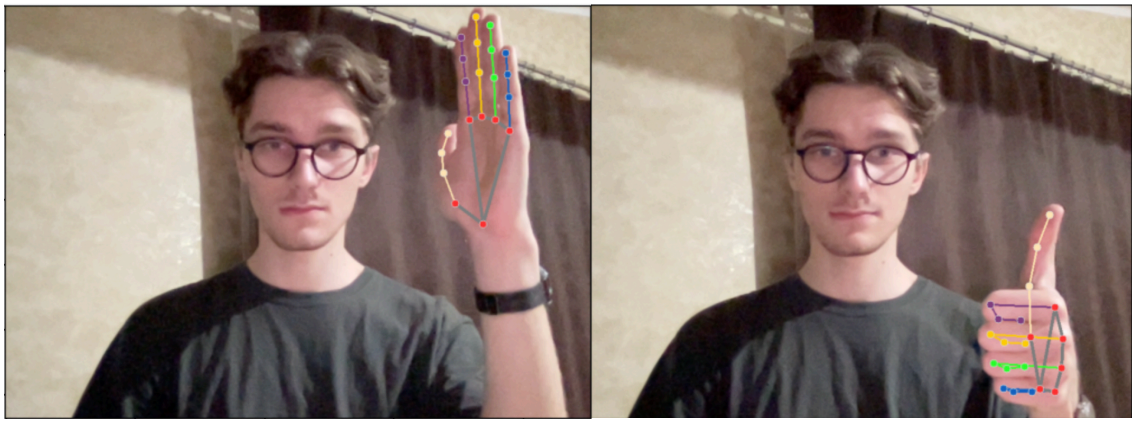


Рисунок 2.5 – Демонстрація детектування руки та характерних точок кісті руки

2.3 Аналіз емоцій учасників

Аналіз емоцій за відеопотіком на якому зображене обличчям не відрізняється суттєво від задач класифікації. У 2013 році Kaggle опублікував завдання з розпізнавання виразу обличчя. Очікувалось, що дослідники

розроблять моделі, які розпізнають сім різних емоцій на обличчях людей. Однак результати того часу не давали значних результатів за точністю оцінки, на відміну від сучасних [27].

Використання набору даних Fec2013 забезпечить управління процесами навчання та оцінювання. Нестиснута форма набору даних вимагає 295 МБ простору, проте стиснута версія займає лише 92 МБ. Набір даних складається з 28 тис. навчальних фотографій і 3 тис. тестових зображень. Кожна фотографія була збережена у вигляді файлу розміром 48×48 пікселів. Сирий набір даних складається з пікселів зображення (48×48 = 2304 значень), емоції кожного зображення та типу використання (як тренувальний або тестовий екземпляр).

В останні роки глибоке навчання домінує в дослідженнях комп'ютерного зору. Для вирішення цієї задачі будуть використані згорткові нейронні мережі. Крім того, ШНМ побудована з використанням Keras та бекенду на TensorFlow.

В таблиці 2.4 описана Head частина моделі детектування емоцій.

Таблиця 2.4 – Head частина ШНМ VGG-16 для розпізнавання емоцій

Шар (тип)	**Вихідна форма**	**Параметри**
Conv2D	(None, 48, 48, 1)	64
MaxPooling2D	(None, 2, 2)	25
Conv2D	(None, 3, 3)	64
Conv2D	(None, 3, 3)	64
AveragePooling2D	(None, 2, 2)	9
Conv2D	(None, 3, 3)	128
Conv2D	(None, 3, 3)	128
AveragePooling2D	(None, 2, 2)	9
Dense	(None, 14, 14, 512)	1024
Dense	(None, 14, 14, 512)	1024

Завдяки цьому можливо точно класифікувати 214 об’єктів як гнів. Однак дев’ять об’єктів були визначені як огидні, коли вони насправді були розлючені.

Розпізнавання емоцій базується на отриманні характерних точок обличчя. Одна з найкращих моделей для вилучення характерник ознак є Face Mesh [28 – 30], яка базується на двох основних етапах: детектування обличчя та вилучення характерних ознак.

В моделі використовувалося навчання з перенесенням для навчання мережі, яка одночасно прогнозує 2D семантичні контури на основі анотованих реальних даних і 3D координати орієнтирів на основі синтезованих даних для 3D орієнтирів обличчя [31].

Мережа 3D-орієнтирів отримує на вхід обрізаний відеокадр без додаткового введення глибини. Модель виводить положення 3D точок, а також ймовірність того, що обличчя присутнє і достатньо вирівняне у вхідних даних. Поширеним альтернативним підходом є прогнозування 2D-теплової карти для кожного орієнтира, але він не піддається прогнозуванню глибини і має високі обчислювальні витрати для такої кількості точок. Точність і надійність моделі підвищується шляхом ітеративного бутстрапінгу і уточнення прогнозів [32]. Таким чином можлива розширити набір даних до все більш складних випадків, таких як гримаси, кут нахилу і оклюзії.

В застосунку до моделі характерних точок обличчя пропонується ще одна модель, яка звертає увагу на семантично значущі області обличчя, а отже, більш точно прогнозує орієнтири навколо губ, очей і райдужної оболонки за рахунок більшої кількості обчислень.

На базі отриманих результатів можливо реалізувати такі задачі як:

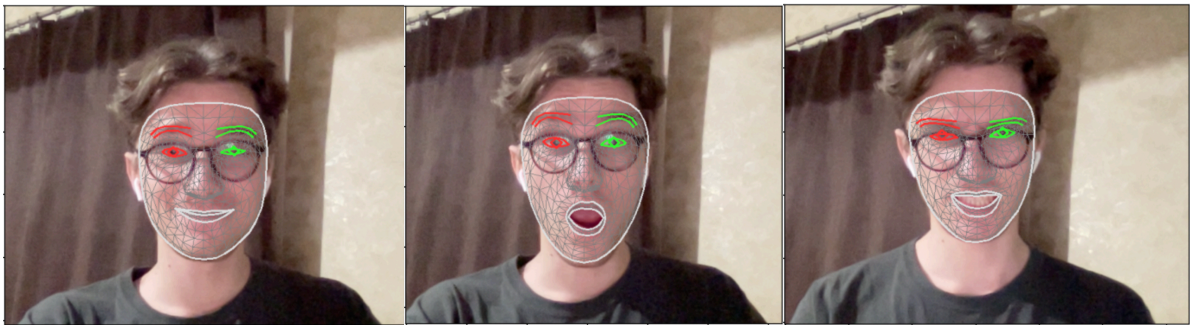
– надягання віртуальної маски;

– детектування емоцій;

– детектування віку, статі або раси;

– анімація 3D або 2D моделей для створення анімаційних відеокліпів.

На рисунку 2.6 зображено приклад отримання характерних точок обличчя та накладання їх на зображення.



(а) (б) (в)

Рисунок 2.6 – Приклад аналізу емоцій та детектування характерних точок: (а) нейтральний; (б) страх; (в) огида

2.4 Визначення методів детектування, верифікації та розпізнавання для використання у системі моніторингу

Для розробки алгоритму моніторингу дій учасників потрібно виділити декілька важних компонент:

– інтеграція з платформою або сервісом відеоконференції для захвату відеопотіку та в разі потреби аудіо доріжки;

– визначити правила, за якими буде працювати система та які потрібно вимагати дотримуватися учасників конференції;

– бєкенд – компонент системи, котрий зможе підтримувати зв’язок між інтерфейсом користувача, сервісом відеоконференції та системами аналізу, аналізувати вихідні та вхідні данні;

– системи аналізу дій учасників, або пайплайн обробки відеопотіку.

В якості платформи відеоконференції в даному досліджені була вибрана Zoom, через її популярність, та велику бібліотеку програмних компонентів, SDK, котрі можуть бути інтегровані з власних програмним забезпеченням.

В якості правил системи були визначені такі правила:

– під час конференції учасник зобов’язується мати стабільне інтернет підключення, камеру із задовільною якістю зображення, та гарно освітлене робоче місце. Якщо обличчя не може бути детектовано на зображенні, або на зображенні завеликий рівень шуму, то дане правило порушене;

– під час конференції в учасника постійно повинна бути увімкнена камера. Якщо відеопотік учасника призупинився в той час коли сесія не вважається закінченою, учасник все ще підключений, та отримуються відеопотіки інших учасників, то дане правило порушено;

– під час конференції в учасника не повинно бути зайвих людей. Якщо на відеопоці під час детектування обличчя помічено більше одного обличчя, то дане правило порушено;

– під час конференції учасник постійно повинен бути в полі зору камера так, щоб обличчя було видно. Якщо на відеопотоці не може бути детектовано жодного обличчя, та відеопотік не припинявся на ще активний, то дане правило порушено.

Бєкенд повинен вирішувати усі питання авторизації та зв’язку клієнтського інтерфейсу з сервісами аналізу даних, передавати данні до сервісів аналізу, отримувати результат, робити висновки, та сповіщати підписників системи о результатах. Крім того він повинен мати доступ до бази даних учасників з еталонними зображеннями та відповідність користувацьких даних до учасників конференції та підтримувати правила конференції. Для цього потрібно декілька компонент:

– підключення в реальному часі до клієнтів;

– систему сповіщення;

– систему передачі відеопотіку до сервісів аналізу;

– базу даних користувацьких даних;

– систему авторизації та аутентифікації з сервісом відеоконференції.

Системи аналізу даних відеопотіку в свою чергу повинні приймати данні від бєкенд сервісів, вміти їх розуміти, вилучати окремі фрейми кожного учасника, проводити передобробку, аналіз та постобробку та передавати назад

до бєкенд сервісів за номером сесії. Серед систем аналізу даних за типом задачі можна виділити декілька типів моделей:

– детектування обличчя;

– верифікація особи;

– збір статистичних даних;

– інше.

Моделі повинні не тільки давати гарний результат вірного прогнозування, а й опрацьовувати дані в реальному часі (декілька кадрів) в секунду приминанні для детектування обличчя, тобто бути швидкою з меншою кількістю скритих шарів.

На підставі дослідження для реалізації застосунку будуть використані моделі:

– FaceNet – для проводження верифікації обличчя;

– MTCNN – для детектування обличчя;

– FaceMesh – для детектування характерних точок обличчя;

– VGG-16 – для детектування емоцій.

3 РЕАЛІЗАЦІЯ МЕТОДУ МОНІТОРІНГУ ДІЙ УЧАСНИКІВ ОНЛАЙН-КОНФЕРЕНЦІЙ

3.1 Налаштування програмного середовища

Для реалізації застосунку під кодовою назвою SmartGuard було вибрано такі технології та мови програмування:

– клієнтський застосунок та інтерфейс: Zoom Meeting SDK Web, JavaScript, TypeScript, Angular 15, як фреймворк побудови SPA застосунків, та інше;

– бєкенд частина: .Net 7, ASP Net Core, SignalR для підтримки realtime connection та інше;

– сервіси аналізу: Python 3.8, Anaconda, OpenCv, Dlib, DeepFace, Pika та інше;

– інше: Redis в якості key-value бази даних та дистрибутивно розподіленого кешу, Docker, RabbitMQ в якості Message Broker, Git в якості системи контролю версій.

Для того щоб запустити застосунок потрібно виконати команди в терміналі або встановити Desktop клієнт за допомогою інтернету самотужки.

Всі наведені команди в таблиці 3.1 використовують Homebrew – система керування пакетів.

Таблиця 3.1 – Команди для підготовки програмного середовища

Команда	**Компонент**	**Опис**
git clone https://github.com/ardasovvadim/SmartGuard.git	Git SmarGuard repo	
brew install --cask anaconda	Anaconda	
conda env create -f environment.yml	Anaconda virtual env	Потрібно виконати в директорії SmartGuar/ SmartGuard.Recognition

Продовження таблиці 3.1

Команда	**Компонент**	**Опис**
brew install docker	Docker	
docker run --name some-redis -d redis	Redis	
docker run -d --hostname my-rabbit --name some-rabbit rabbitmq:3	RabbitMQ	
python main.py -c faceAnalysing	Статистичний компонент	Потрібно виконати в директорії SmartGuar/ SmartGuard.Recognition
python main.py -c faceVerifying	Компонент верифікації	Потрібно виконати в директорії SmartGuar/ SmartGuard.Recognition
python main.py -c faceDetecting	Компонент детектування	Потрібно виконати в директорії SmartGuar/ SmartGuard.Recognition
brew install dotnet	dotnet SDK	
dotnet run	Бєкенд компонент	Перед виконанням команди потрібно вірно вказати параметри в appsettings.json. Потрібно виконати в директорії SmartGuard
brew install node	Встановити Node JS	
npm i && npm run start	Клієнтська частина	Потрібно виконати в директорії SmartGuard/Web

Після останньої команди SmartGuard готовий для налаштування та використання в Development версії.

Zoom пропонує обширну бібліотеку та набір SDK для створення інтеграцій для власних застосунків.

Порівняння Zoom SDKs наведено в таблиці 3.2.

Таблиця 3.2 – Порівняння Zoom SDKs

SDK	Платформа	Мова програмування	Zoom meetings	Raw video	Raw audio	Складність реалізації
Meeting SDK	Android	Java	Так	+	+	2
	iOS/macOS	Objective-C		+	+	2
	Web	JavaScript		-	-	1
	Windows	C++ / C#**		-	-	3
Video SDK	Android	Java	Ні*	+	+	3
	iOS/macOS	Objective-C		+	+	3
	Web	JavaScript		-	-	2
	Windows	C++		+	+	3

* – в Video SDK [33] не можливо використати стандартні програмні засоби, які доступні для кожної платформи та з'єднуватися зі звичайними Zoom meetings, замість цього використовуються «ізольовані» конференції, які проходять через сервери Zoom: обробка мультимедіа, запис, livestream, але користувацький інтерфейс потрібно писати самотужки.

** – для Meeting SDK [34] Windows платформи доступний API на C++, але є можливість використовувати сучасний засоби за допомогою мови програмування C#. Meeting SDK C# є врапером або обгорткою над C++ API, більшість речей вже не підтримуються як і сам репозиторій.

Для дослідження було вибрано Zoom Meeting SDK для платформи Web, через значну швидкість та простоту розробки, гнучкість, бо Web клієнт може використовуватися на будь якому пристрої з браузером, підтримкою JavaScript та SharedBuffer [35] технології та виходом до інтернету. Крім того в Meeting SDK є можливість користуватися звичайними, безкоштовними відео

конференціями та клієнтами Zoom доступними на кожній платформі: Windows, MacOS, Android, iOS.

Значним недоліком Meeting SDK Web в тому, що не має підтримки Raw Video/Audio Recording технології, котра дає змогу отримати необроблений відео та аудіо потік кожного учасника конференції окремо. Але це питання буде вирішено захопленням відеопотіку з Web Client.

Zoom Meeting SDK на платформі Web – це бібліотека, або програма з API на JavaScript, яка дозволяє інтегрувати Zoom у власне бізнес рішення, що підходить для вирішення даної проблеми.

Для того щоб підключитися до відеоконференції потрібно виконати декілька кроків:

– загрузити Wasm бібліотеку;

– загрузити Web SDK;

– загрузити файл локалізації;

– ініціалізувати бібліотеку;

– отримати JWT токен;

– під’єднатися до відео конференції.

Для отримання JWT [36] токен, було використаний власний API.

Після цих кроків буде отримано звичний Zoom інтерфейс користувача дуже схожий на Desktop платформу. Zoom використовує технологію WebAssembly базового функціоналу та WebSockets для комунікації, через це відео формується на сервері та відображається в canvas HTML елементі, та не виводиться в кожний окремий video елемент для кожного учасника конференції, та являю собою зіставлене зображення з усіх фреймів.

Для функцій drag and drop, текстової інформації, Zoom передає положення video frame кожного учасника відносно спільного canvas елемента, який і відображає відеопотік, та будує рамку за допомогою JavaScript. Через це, якщо змінити лейаут або макет застосунку, то Zoom вимагає затримку на змінення положення генерації вихідного елементу.

Для досягнення поставленої мети та вилучення відеопотіку кожного учасника окремо, було використано той самий підхід:

– за таймером при необхідності був знятий спільний фрейм з canvas, де зображені всі учасники;

– за допомогою запитів до document об’єкта було вилучено відносне положення рамки зображення кожного учасника окремо.

Таким чином на виході маємо спільний фрейм та мета інформація – положення, кожного учасника на зображенні відносно спільного зображення. Все що залишається, це виділити кожне окреме зображення за даною інформацію. Таким чином був отриманий доступ до відеопотіку кожного учасника відеоконференції окремо. На рисунку 3.1 схематично зображений алгоритм отримання відеопотіку учасників Zoom конференції.

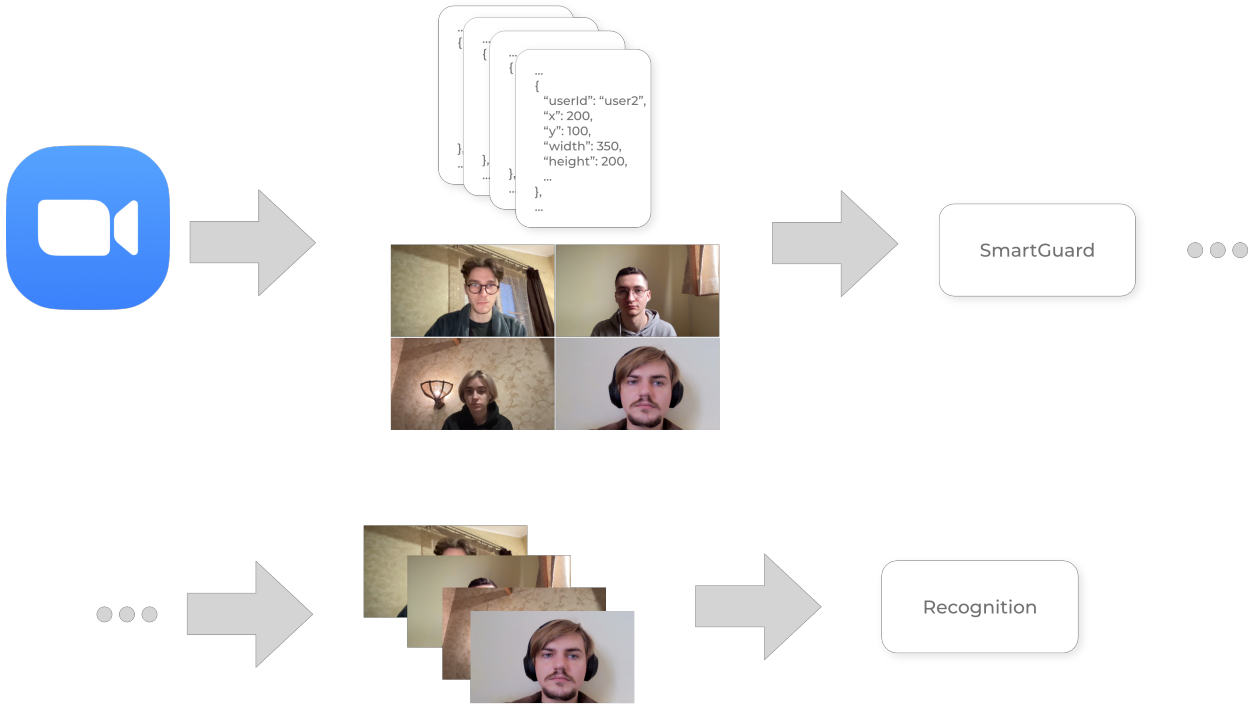


Рисунок 3.1 – Схематичний алгоритм отримання відеопотіку учасників з конференції Zoom

Запропонований підхід допомагає вирішити декілька проблем – це вилучити відеопотік, кожного учасника конференції окремо, та зменшує час на обробку та аналіз.

3.2 Проєктування бази даних для зберігання результатів моніторингу

Для розробки прототипу застосунку SmartGuard було використано key-value базу даних Redis.

Redis – це Remote Dictionary Server або сервер, який зберігає свої дані в оперативній пам’яті, що реалізує розподілену базу даних ключ-значення в пам’яті з необов’язковою довговічністю. Redis [37] забезпечує ефективність використання пам’яті, високу швидкість роботи, доступність і надає деякі функції, такі як реплікація, кластеризація тощо.

Redis не має зовнішніх ключів, підтримки цілісності даних, тригерів або SQL підтримки, але це і не знадобиться для даного дослідження. Значний плюс цього вибору – це швидкодія, що є найважливішим умовою обробки відеопотіку в реальному часі різними моделями штучного інтелекту (ШІ) та великої кількості читання та запису даних до бази даних.

Основними моделями системи є:

– FrameMessage;

– FrameResultMessage;

– Alert;

– ZoomSession;

– StatisticInfo;

– VerifyingInfo;

– DetectingInfo;

– SgAttendee.

Детальніше частина моделей системи описано в таблицях 3.3 – 3.10.

Таблиця 3.3 – Модель SgAttendee

Назва	**Тип даних**	**Додаткова інформація**
userId	string	Zoom ідентифікатор
isHost	bool	Господар конференції
verified	bool?	
lastVerified	bool?	

Продовження таблиці 3.3

Назва	**Тип даних**	**Додаткова інформація**
lastVerifiedTime	DateTime?	Останній час верифікації
emotion	string?	Домінуюча емоція
age	int?	
genre	string?	
race	string?	

Таблиця 3.4 – Модель даних FrameMessage

Назва	**Тип даних**	**Додаткова інформація**
FrameId	string	
SessionId	string	
Actions	string[]	Додаткова інформація об actions для сервісів аналізу
Attendees	AttendeeFrame[]	Список учасників для обробки

Таблиця 3.5 – Модель даних AttendeeFrame

Назва	**Тип даних**	**Додаткова інформація**
userId	string	
username	string	
height	int	Висота зображення відносно фрейму
width	int	Ширина зображення відносно фрейму
x	int	Ліва верхня координата абсцис зображення відносно фрейму
y	int	Ліва верхня координата ординат зображення відносно фрейму
size	int	
time	DateTime?	
additionInfo	string?	

Таблиця 3.6 – Модель даних Alert

Назва	**Тип даних**	**Додаткова інформація**
text	string	
color	int	Колір сповіщення
code	int	Код сповіщення
data	object?	
time	DateTime	Час створення
frameId	string?	

Таблиця 3.7 – Модель даних ZoomSession

Назва	**Тип даних**	**Додаткова інформація**
sessionId	string	SmartGuard ідентифікатор
connectionId	string	SignalR ідентифікатор
createdTime	DateTime	

Таблиця 3.8 – Модель даних StatisticInfo

Назва	**Тип даних**	**Додаткова інформація**
emotion	string	Json
age	int	
genre	string	
race	string	Json

Таблиця 3.9 – Модель даних VerifyingInfo

Назва	**Тип даних**	**Додаткова інформація**
verified	bool	
distance	int	Відстань між векторами
threshold	int	Поріг прийняття рішення
model	string	Використана модель
detector_backend	string	
similarity_metric	string	

Таблиця 3.10 – Модель DetectionInfo

Назва	**Тип даних**	**Додаткова інформація**
frameId	string	Ідентифікатор фрейму
userId	string	

Продовження таблиці 3.10

Назва	**Тип даних**	**Додаткова інформація**
x	int	
y	int	
width	int	Ширина фрейму
height	int	Висота фрейму
skipped	bool?	
multiplePersonDetected	bool?	
personIsNotDetected	bool?	
lastStatisticUpdate	DateTime?	
active	bool?	
joinedTime	DateTime?	Час приєднання
joinedTimes?	DateTime[]	
leftTime	DateTime?	Час відключення
leftTimes	DateTime[]	

Дані постійно обновляються, додаються та переміщуються, тому використання Redis, як бази даних на час активної сесії більш ніж обґрунтовано.

3.3 Проєктування сервісу

Основними компонентами системи є:

– SmartGuard.Web – клієнтський інтерфейс або web застосунок. Головна частина застосунку, з якою взаємодії користувач. Web частина спілкується з WebApi компонентом та Zoom;

– SmartGuard.WebApi – бєкенд частина, яка є зв’язуючим компонентом між SmartGuard.Web та SmartGuard.Recognition. Представлена в виді REST API, містить в собі хаб для отримання realtime connection, та background сервісів, які спілкуються з сервісами аналізу даних, проводить постобробку результатів та підтримує правила сесії;

– SmartGuard.Recognition – набір сервісів для аналізу відеопотіку: detection, verify, statistic. Обробляють потік даних за пакетами даних (фреймами) з прикріпленою до неї мета інформації, після процесів аналізу відправляють результат назад в брокер повідомлень. Кожен окремий сервіс повністю ізольований один від одного та запускається паралельно. Це дозволяє виконувати задачі по аналізу паралельно не чекаючи результату виконання іншого. Крім того такий відхід дозволяє швидко та зручно збільшувати кількість сервісів одного типу для звеличення пропускної здібності;

– RabbitMQ – брокер повідомлень [38]. Реалізує чергу повідомлень та гарантує, що повідомлення буде оброблено лише одним споживачем (consumer);

– Zoom – сервіс відеоконференції. Відповідальний за з'єднання членів конференції, створення сесій, менеджмент учасників та базовий функціонал: синхронізація між календарями, стенографія, відео запис конференції, живий чат тощо.

Схема взаємодії між компонентами системи зображена на рисунку 3.2.

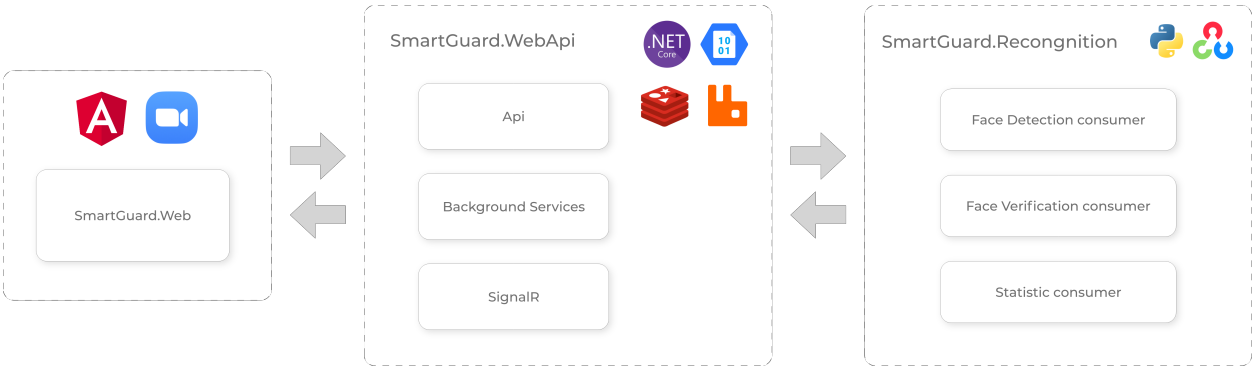


Рисунок 3.2 – Архітектура SmartGuard

Запропонована архітектура дає змогу забезпечити відмовостійкість та масштабованість, тобто кожний окремий сервіс бути розташований в власному ізольованому docker контейнері, що дозволить збільшувати або навпаки зменшувати кількість сервісів залежно від кількості сесій та

навантаження системи. Крім того, якщо станеться невиправна помилка в одному з них, то це не буде сприяти роботі інших сервісів.

Наступним кроком буде спроєктувати послідовність дій актора або користувача з системою та її частинами.

На рисунку 3.3 зображена UML діаграма послідовності [39] дій актора та компонентів системи SmartGuard. Діаграма описує взаємодію користувача від запуску системи, створення сесії та приєднання для конференції, до запуску процесу аналізу відеопотіку.

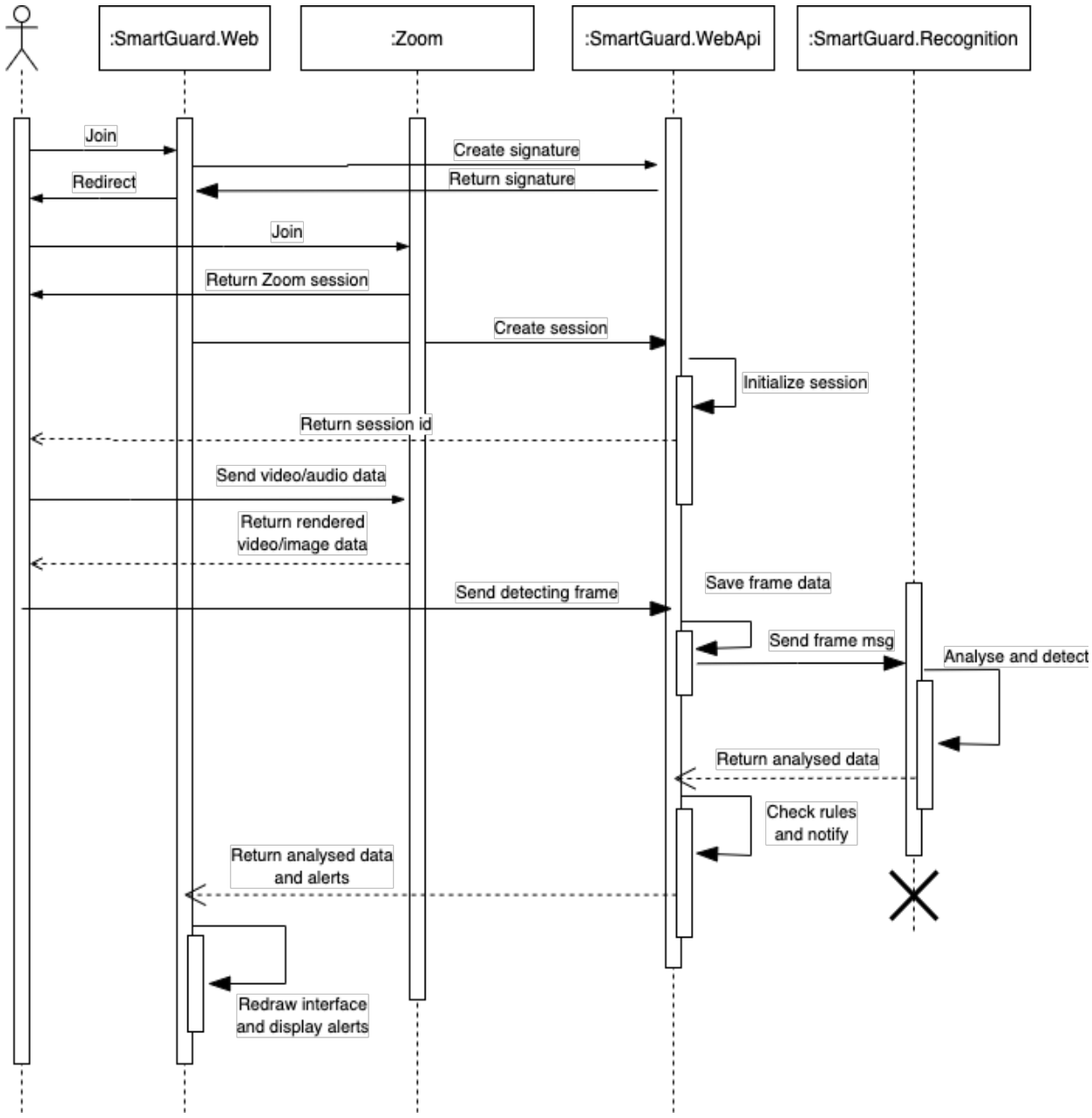


Рисунок 3.3 – Діаграма послідовності взаємодії компонентів системи

3.4 Реалізація окремих функцій та ілюстрація роботи

Основним компонентом з яким взаємодіє користувач – це SmartGuard.Web. Застосунок спрямований для викладача, комісії, або для адміністрації, яка хоче стежити за правилами та їх дотримання під час відео конференції.

Початковим вікном є вікно приєднання до конференції (рис. 3.4).

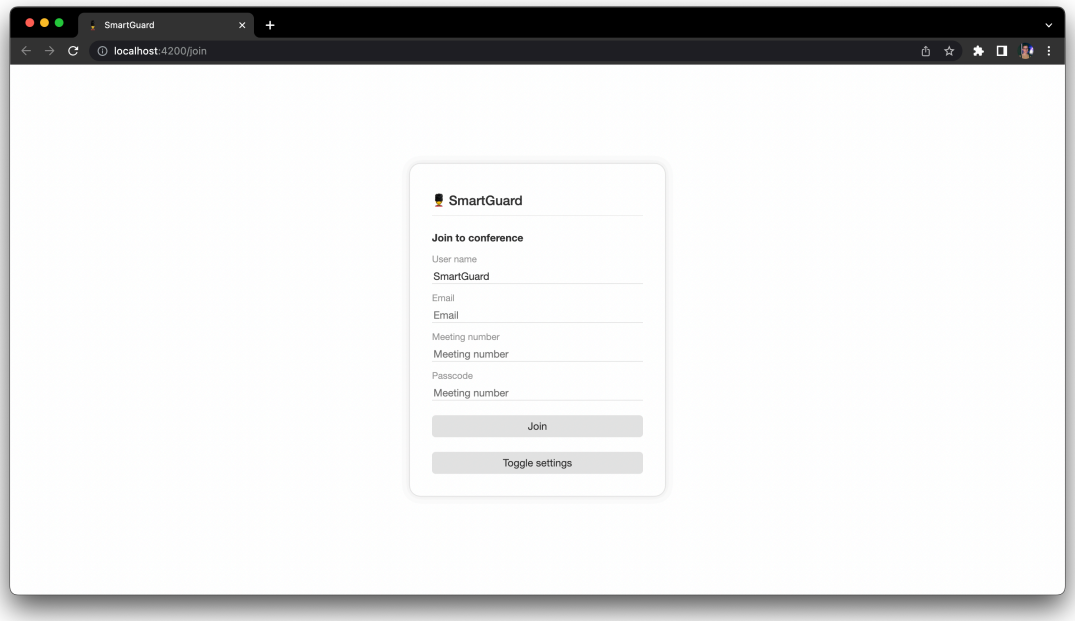


Рисунок 3.4 – Початкове вікно Join to conference

Користувач має змогу ввести такі дані як ім’я користувача, email, номер конференції та пароль для підключення.

В момент підключення до системи SmartGuard повинен перевірити правильність вводу даних. Сповістити користувача в разі виникнення помилки, наприклад, що сесія за даним кодом не існує, або пароль від кімнати конференції не співпадає [40].

Повідомлення буде виведено в якості сповіщення над формою вводу даних.

Крім того користувач має змогу змінити налаштування сесії, правила проведення до підключення (рис. 3.5).

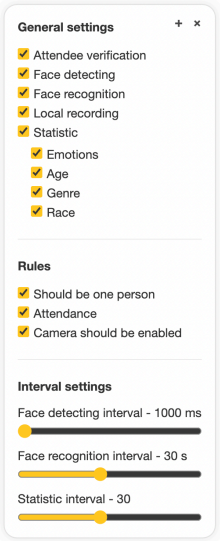


Рисунок 3.5 – Налаштування сесії

Після того як користувач налаштував сесію та ввів усі потрібні дані, він може успішно перейти до конференції по кліку на кнопку «Join».

На рисунку 3.6 зображено наступне вікно – налаштування камери та мікрофона перед підключенням до Zoom.

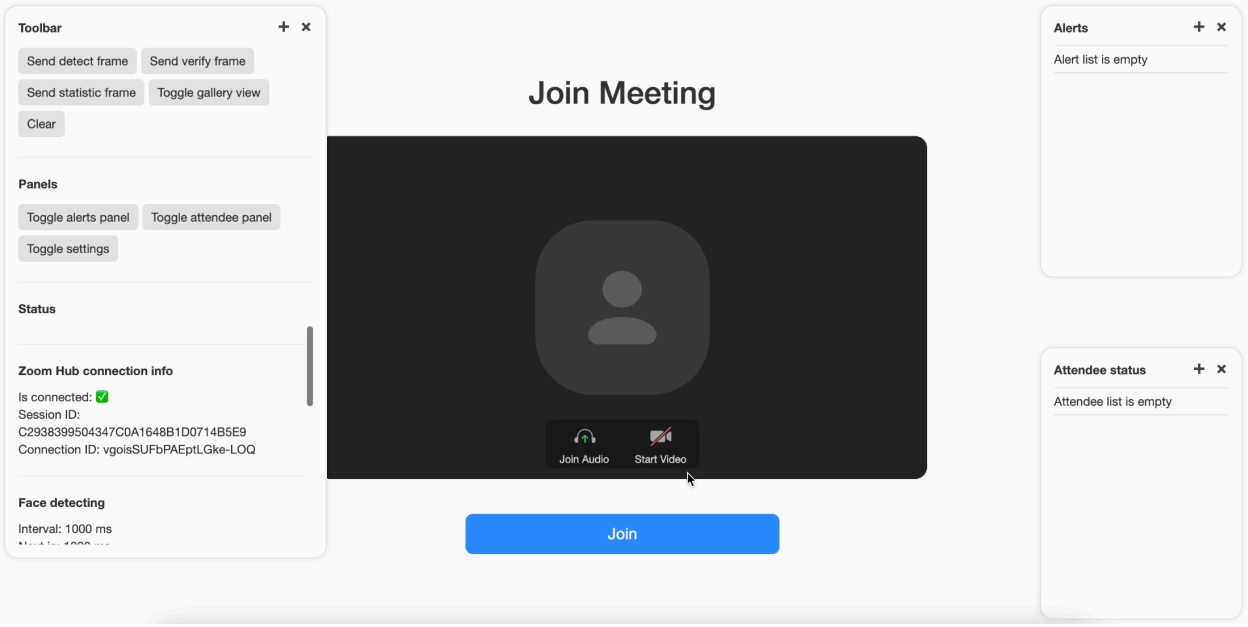


Рисунок 3.6 – Вікно налаштування камери та мікрофона

Коли налаштування камери та мікрофона успішно завершені, користувач може натиснути на наступну кнопку «Join» для переходу до звичайного Zoom інтерфейсу з SmartGuard інтеграцією (рис. 3.7).

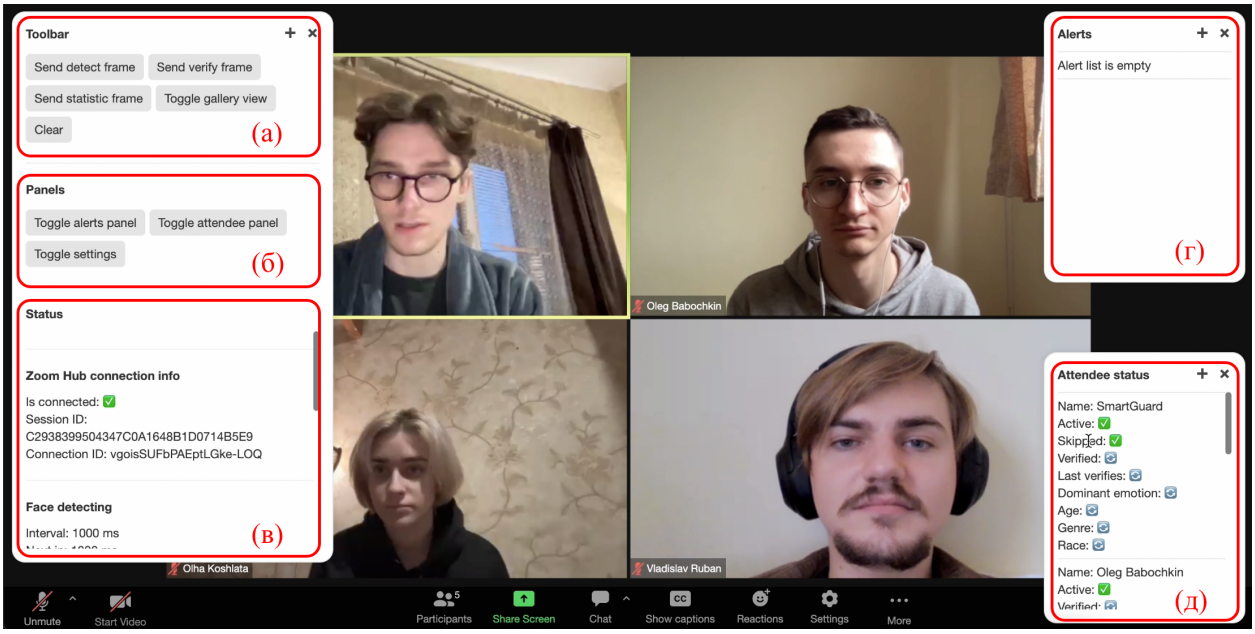


Рисунок 3.7 – Інтерфейс користувача SmartGuard під час сесії:
(а) кнопки для ручного управління; (б) активація допоміжних вікон;
(в) показники системи; (г) повідомлень системи;
(д) вікно статусу учасників відеоконференції

Завдяки різним вікнам на кнопкам, користувач може контролювати правила дотримання сесії та змінювати їх під час проходження сесії.

Toolbar секція відповідає за кнопки ручного управління, якщо йому потрібно виконати якусь операцію миттєво, а не по проходженню таймер.

Panels секція відповідає за контроль відкритих вкладок. Завдяки цьому вікну можна відкривати та закривати вкладки Attendee status, Alerts, Settings.

Status секція відповідає за звітування статусу різних таймерів та статус підключення до системи, також є можливість почати або зупинити таймер аналізу окремо.

Alerts вікно відповідає за відображення в реальному часі повідомлень: успішність верифікації учасників, порушення правил, час оновлення статистичних даних або помилки, якщо вони виявлені.

Attendee status вікно відповідає за відображення статусу учасників конференції: ім’я, активність, статус верифікації, час верифікації, емоції, розпізнаний вік, стать та расу.

3.4.1 Розпізнавання обличчя учасника онлайн конференції

Для запуску процесу детектування, розпізнавання обличчя та збору статистичних даних повинні бути запущені сервіси або таймери «Face detection», «Face verification», «Statistic» відповідно, крім того є можливість здійснення операцій без таймера в ручну по натисканню на кнопки «Send detection frame», «Send verify frame», «Send statistic frame» відповідно.

Процес детектування, розпізнавання та збору статистичної інформації зображено відповідно на рисунках 3.8 – 3.11.

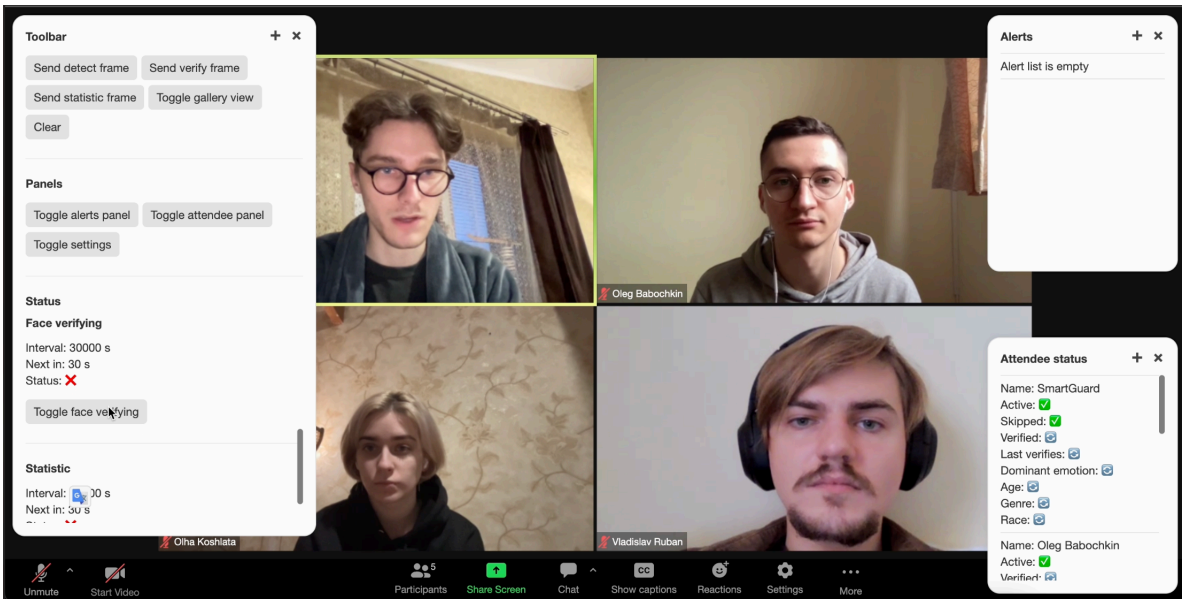


Рисунок 3.8 – Процес детектування обличчя (фрейм до оброки)

Рисунок 3.9 – Процес детектування обличчя (фрейм після обробки)

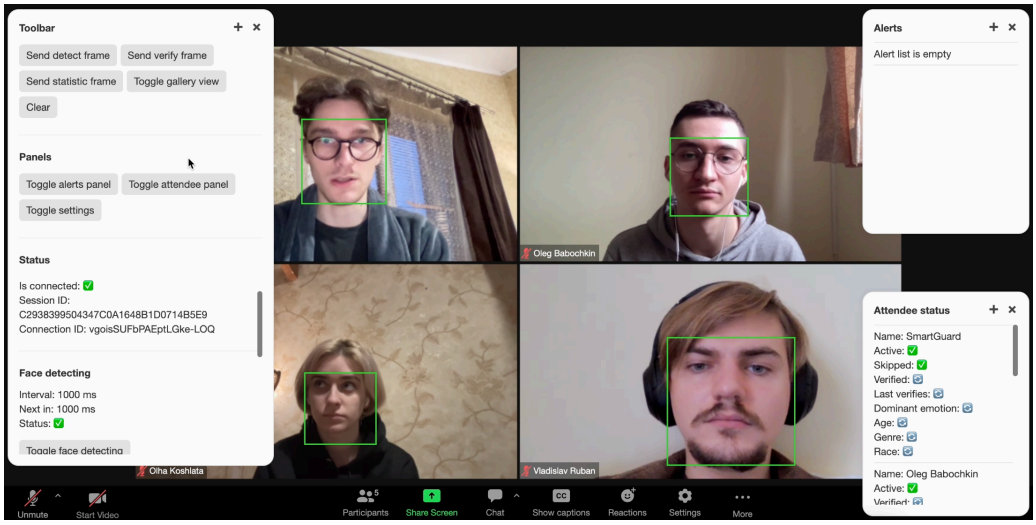


Рисунок 3.10 – Процес верифікації обличчя (до аналізу)

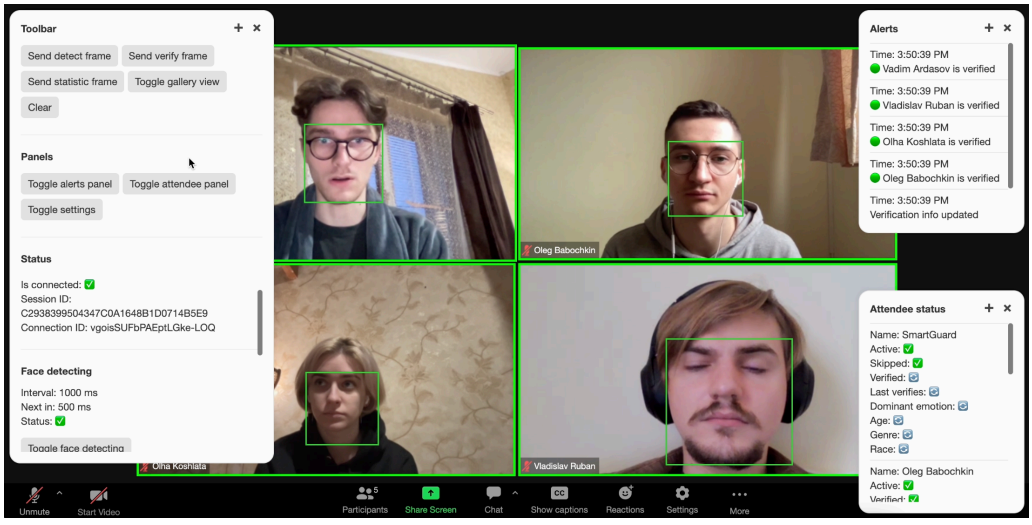


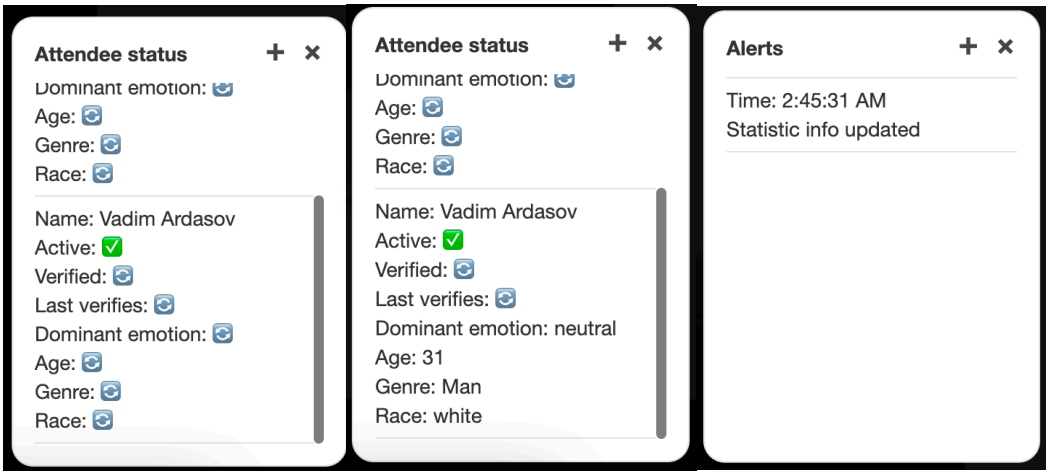
Рисунок 3.11 – Процес верифікації обличчя (після аналізу)

На рисунку 3.12 інтерфейс підсвічує зміни після процесу верифікації:

– рамка учасника змінює колір на зелений, якщо він пройшов процес верифікації, в іншому випадку на червоний;

– відповідне повідомлення з’являється в Alerts вікні;

– інформація щодо учасника обновляється в вікні Attendee status.



(а) (б) (в)

Рисунок 3.12 – Процес збору статистичних даних:
(а) інформація до обробки фрейму; (б) оновлена інформація після аналізу;
(в) демонстрація повідомлення об оновлених даних

3.4.2 Моніторинг дій під час конференції

Моніторинг дій виконується в реальному часі відштовхуючись від наданої інформації:

– список учасників конференції;

– вихідного відеопотіку з Web застосунку;

– результатів аналізу фреймів;

– результату попередніх фреймів на статусу системи та учасників.

Системою підтримуються такі правила проведення сесії:

– під час конференції в учасника постійно повинна бути увімкнена камера;

– під час конференції в учасника не повинно бути зайвих людей;

– під час конференції учасник постійно повинен бути в полі зору камери.

Усі проаналізовані дані, повідомлення о порушенні або успішному процесі зберігаються для подальшого їх аналізу або перегляду користувачем.

На рисунках 3.13, 3.14 зображено процес підтримки правила присутності учасника в полі зору камери. На рисунку 3.13 Oleh Babochkin відсутній в полі зору камеру тому він виділений червоною рамкою та відповідне повідомлення сформоване в Alerts вікні.

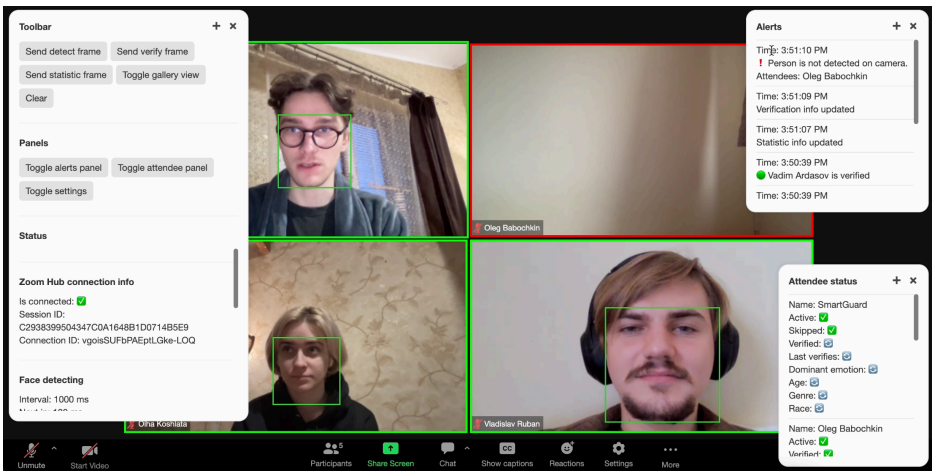


Рисунок 3.13 – Приклад підтримки правила присутності в полі зору камери

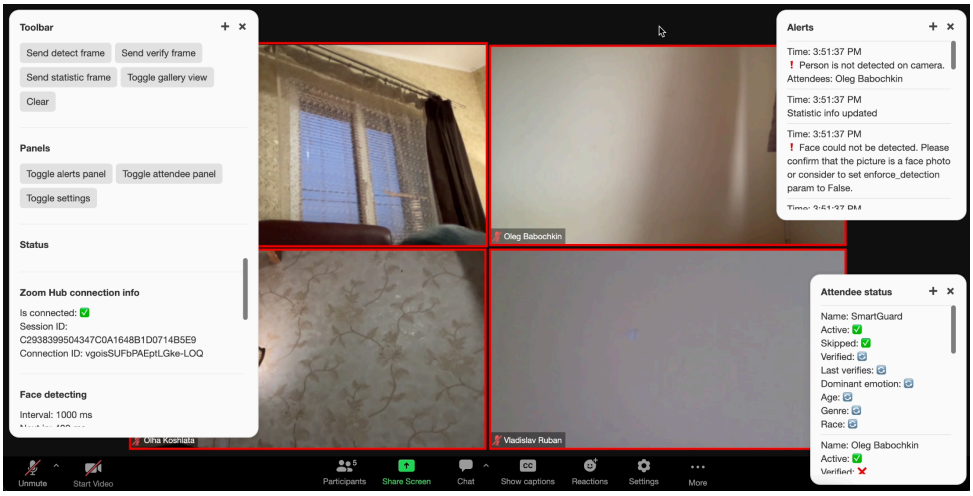


Рисунок 3.14 – Приклад підтримки правила присутності в полі зору камери (усі учасники відсутні)

На рисунку 3.15 зображено процес підтримки правила верифікації учасника конференції. Olha Koshlata знаходиться на місці Vadim Ardasov, що заперечить правилам проходження сесії, тому вона виділена червоною рамкою, та відповіде повідомлення сформоване в Alerts вікні.

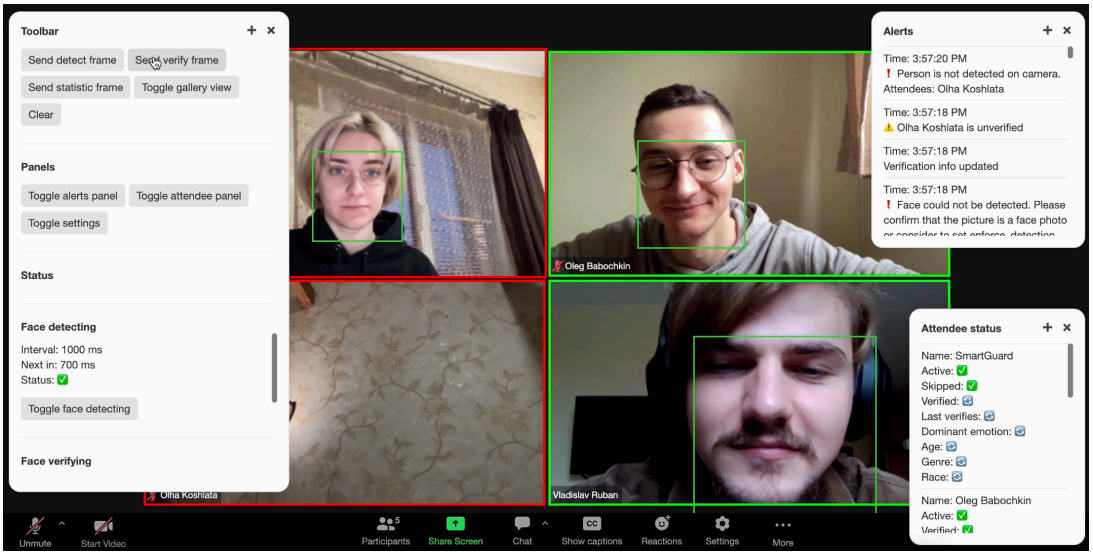


Рисунок 3.15 – Приклад підтримки правила верифікації

На рисунку 3.16 зображено процес підтримки правила, що тільки одна людина може бути присутня в кадрі. В фреймі Vadim Ardasov зафіксовано дві людини, що суперечить правилам сесії, тому Vadim Ardasov виділений червоною рамкою, та відповіде повідомлення сформоване в Alerts вікні.

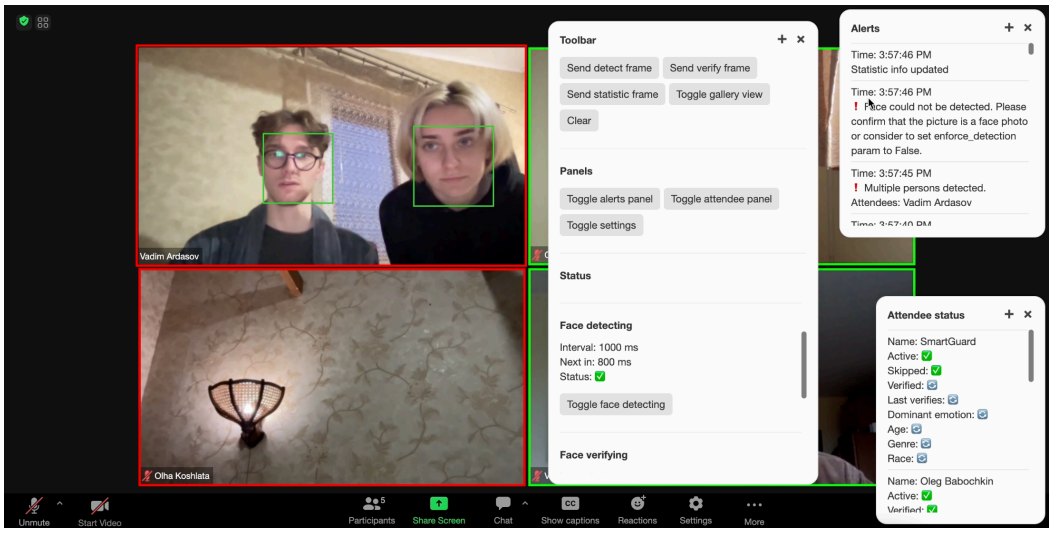


Рисунок 3.16 – Приклад підтримки правила одної людини в кадрі

3.5 Візуалізація результатів моніторингу

Як було згадано вище, усі результати аналізу сервісів, правопорушень або успішної верифікації збережено для подальшого аналізу та перегляду.

Після завершення сесії відеоконференції, користувач переходить на сторінку з вікном вибору дії:

– приєднатися до іншої конференції;

– перейти до статистики;

– зберегти відео запис конференції.

На рисунку 3.17 зображено вікно завершення відео конференції. Для переходи до сторінки статистики користувач повинен натиснути на кнопку «Go to statistic».

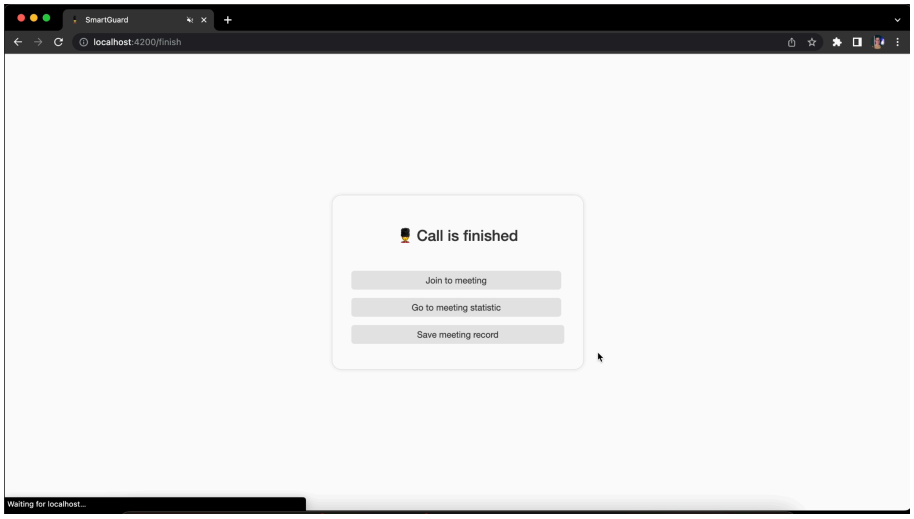


Рисунок 3.17 – Вікно завершення відео конференції

Інтерфейс статистики має таби для переходу в різні вікна перегляду статистичних даних та кнопки дій.

Відділ статистики має 3 основних вікна:

– Alerts list – таблиця повідомлень, які були зібрані під час конференції;

– Emotions chart – графік залежності кількості учасників від часу за певним емоційним класом;

– Events timeline – графік часових проміжків від повідомлень.

Інтерфейс статистичних даних зображено на рисунку 3.18.

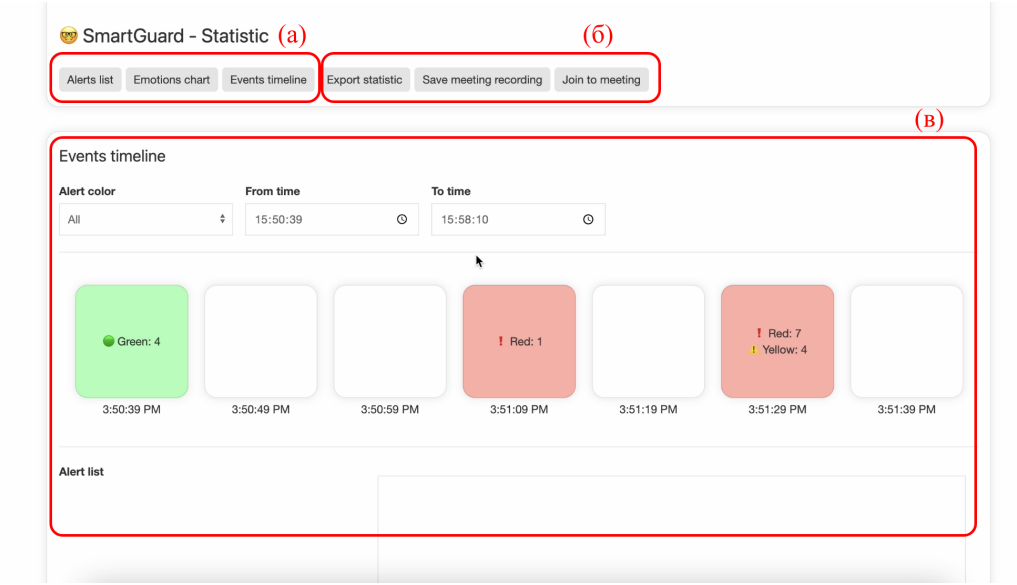


Рисунок 3.18 – Інтерфейс статистичних даних:
(а) вкладки вікон; (б) кнопки дій; (в) вікно статистичних даних

Events timeline вікно зображено на рисунку 3.19. Вікно дозволяє фільтрувати повідомлення зо кольором, проміжком часу, продивитися повідомлення за певний проміжок часу, та по кліку на повідомлення відобразиться закріплений фрейм, який був проаналізований.

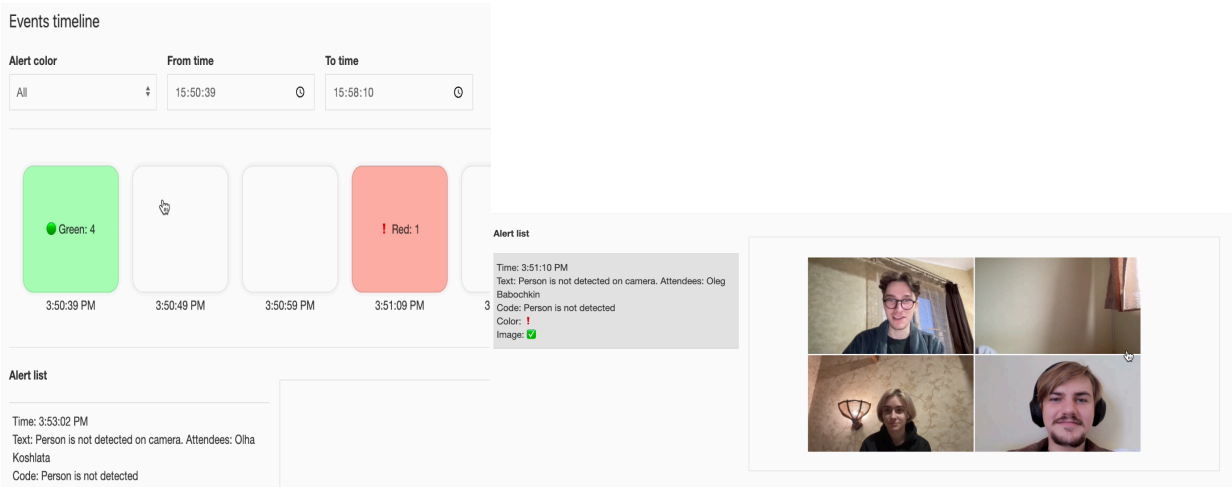


Рисунок 3.19 – Events timeline вікно

На рисунку 3.20 зображено вікно Alerts list. Вікно дозволяє фільтрувати повідомлення за кольором, кодом, текстом, на часом.

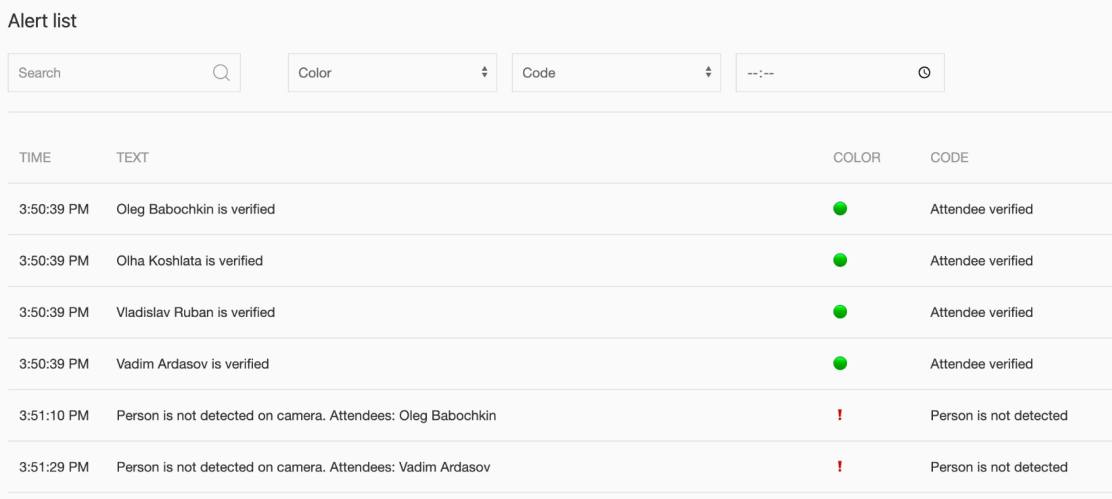


Рисунок 3.20 – Alerts list

Останнім вікном є Emotions chart, який зображений на рисунку 3.21. Графік відображає емоції, які були проаналізовані з фрейму відносно часу проходження сесії відеоконференції.

Рисунок 3.21 – Emotions chart

На підставі отриманих даних можливо роботи наступні висновки, наприклад:

– який процент студентів або учасників конференції були задоволені її проходженням;

– який процент студентів позитивно відреагував та ту чи іншу інформацію;

– який процент студентів спить під час сесії або взагалі виглядить не зацікавленим;

– отримати статистику відвідування конференцій, наприклад, час запізнення та кількість відключень від сесії.

Отримані дані та графіки допоможуть спростити та покращити досвід пошуку підозрілих моментів та моніторингу дій учасників конференції.

Крім аналізу та перегляду даних система надає можливості зберегти статистику в форматі JSON для того, щоб провести свій власний бізнес аналіз за даними, які надає SmartGuard.

ВИСНОВКИ

У рамках кваліфікаційної роботи був розроблений і реалізований метод моніторингу дій учасників онлайн конференції на основі аналізу відеопотіку в рамках програмного застосунку SmartGuard.

В ході виконання роботи було досліджено та вирішено теоретичні та практичні питання:

– ознайомлення та порівняння сервісів онлайн-конференцій;

– впровадження та інтеграція сервісів онлайн-конференцій для програмного застосунку;

– доступ до відеопотіку кожного учасника конференції;

– ознайомлення, вивчення та впровадження нейронних мереж для задачі детектування обличчя;

– ознайомлення, вивчення та впровадження нейронних мереж для задачі верифікації обличчя;

– ознайомлення, вивчення та впровадження нейронних мереж для задачі класифікації емоцій, віку, статі та раси;

– дослідження та розробка методу моніторингу дій учасників за відеопотіком конференції;

– розробка, проєктування програмного застосунку для реалізації методу моніторингу дій учасників;

– проєктування бази даних для зберігання результатів конференції;

– розробка методу аналізу отриманих даних записаних під час сесії відео конференції та їх відображення для подальшого аналізу.

Результати дослідження апробовано у вигляді 2 тез доповідей під час XXXVII Міжнародної науково-практичної конференції «Modern ways of solving the latest problems in science» [41], Восьмої міжнародної науково-технічної конференції «Інформатика, управління та штучний інтелект» [42].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Kumar, A., Kaur, A., & Kumar, M. (2019). Face detection techniques: a review. Artificial Intelligence Review, 52(2), 927-948.

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

3. Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. Journal of cognitive neuroscience, 3(1), 71-86.

4. Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). Deep face recognition.

5. Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 815-823).

6. Learned-Miller, E., Huang, G. B., RoyChowdhury, A., Li, H., & Hua, G. (2016). Labeled faces in the wild: A survey. In Advances in face detection and facial image analysis (pp. 189-248). Springer, Cham.

7. Jung, S. G., An, J., Kwak, H., Salminen, J., & Jansen, B. J. (2017). Inferring social media users' demographics from profile pictures: A Face++ analysis on Twitter users.

8. Bradski, G., & Kaehler, A. (2000). OpenCV. Dr. Dobb's journal of software tools, 3, 120.

9. Deng, J., Guo, J., An, X., Zhu, Z., & Zafeiriou, S. (2021). Masked face recognition challenge: The insightface track report. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 1437-1444).

10. Ketkar, N. (2017). Introduction to keras. In Deep learning with Python (pp. 97-111). Apress, Berkeley, CA.

11. Goldsborough, P. (2016). A tour of tensorflow. arXiv preprint arXiv:1610.01178.

12. Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., ... & Grundmann, M. (2019). Mediapipe: A framework for building perception pipelines. arXiv preprint arXiv:1906.08172.

13. King, D. E. (2009). Dlib-ml: A machine learning toolkit. The Journal of Machine Learning Research, 10, 1755-1758.

14. Padilla, R., Costa Filho, C. F. F., & Costa, M. G. F. (2012). Evaluation of haar cascade classifiers designed for face detection. World Academy of Science, Engineering and Technology, 64, 362-365.

15. Xiang, J., & Zhu, G. (2017, July). Joint face detection and facial expression recognition with MTCNN. In 2017 4th international conference on information science and control engineering (ICISCE) (pp. 424-427). IEEE.

16. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). Ssd: Single shot multibox detector. In European conference on computer vision (pp. 21-37). Springer, Cham.

17. Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., ... & Zhang, Z. (2015). Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. arXiv preprint arXiv:1512.01274.

18. Chen, H. Y., & Su, C. Y. (2018, September). An enhanced hybrid MobileNet. In 2018 9th International Conference on Awareness Science and Technology (iCAST) (pp. 308-312). IEEE.

19. Stringa, L. (1993). Eyes detection for face recognition. Applied Artificial Intelligence an International Journal, 7(4), 365-382.

20. Qawaqneh, Z., Mallouh, A. A., & Barkana, B. D. (2017). Deep convolutional neural network for age estimation based on VGG-face model. arXiv preprint arXiv:1709.01664.

21. Peterson, L. E. (2009). K-nearest neighbor. Scholarpedia, 4(2), 1883.

22. Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., & Scholkopf, B. (1998). Support vector machines. IEEE Intelligent Systems and their applications, 13(4), 18-28.

23. Baltrusaitis, T., Zadeh, A., Lim, Y. C., & Morency, L. P. (2018, May). Openface 2.0: Facial behavior analysis toolkit. In 2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018) (pp. 59-66). IEEE.

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

25. Kölsch, M., & Turk, M. A. (2004, May). Robust Hand Detection. In FGR (Vol. 4, pp. 614-619).

26. Zhang, H., Chang, H., Ma, B., Shan, S., & Chen, X. (2019). Cascade retinanet: Maintaining consistency for single-stage object detection. arXiv preprint arXiv:1907.06881.

27. Goldman, A. I., & Sripada, C. S. (2005). Simulationist models of face-based emotion recognition. Cognition, 94(3), 193-213.

28. Grishchenko, I., Ablavatski, A., Kartynnik, Y., Raveendran, K., & Grundmann, M. (2020). Attention mesh: High-fidelity face mesh prediction in real-time. arXiv preprint arXiv:2006.10962.

29. Ranjan, A., Bolkart, T., Sanyal, S., & Black, M. J. (2018). Generating 3D faces using convolutional mesh autoencoders. In Proceedings of the European conference on computer vision (ECCV) (pp. 704-720).

30. Ansari, A. N., Abdel-Mottaleb, M., & Mahoor, M. H. (2007). 3D face mesh modeling from range images for 3D face recognition. In 2007 IEEE International Conference on Image Processing (Vol. 4, pp. IV-509). IEEE.

31. Yakovleva, O., & Nikolaieva, K. (2020). Research Of Descriptor Based Image Normalization And Comparative Analysis Of SURF, SIFT, BRISK, ORB, KAZE, AKAZE Descriptors. Advanced Information Systems, 4(4), 89-101.

32. А.Р. Ковтуненко, О.В. Яковлева, В.А. Любченко, & О.В. Янголенко (2020) Дослідження сумісного використання математичної морфології та згорткових нейронних мереж для вирішення задачі 64 розпізнавання цінників. Вісник Національного технічного університету ХПІ (3). 24-31.

33. Bartindale, T., Chen, P., Marshall, H., Pozdniakov, S., & Richardson, D. (2021, October). ZoomSense: A Scalable Infrastructure for Augmenting Zoom. In Proceedings of the 29th ACM International Conference on Multimedia (pp. 3771-3774).

34. Kohnke, L., & Moorhouse, B. L. (2022). Facilitating synchronous online language learning through Zoom. Relc Journal, 53(1), 296-301.

35. Jangda, A., Powers, B., Berger, E. D., & Guha, A. (2019). Not So Fast: Analyzing the Performance of {WebAssembly} vs. Native Code. In 2019 USENIX Annual Technical Conference (USENIX ATC 19) (pp. 107-120).

36. Jones, M., Campbell, B., & Mortimore, C. (2015). Json web token (jwt) profile for oauth 2.0 client authentication and authorization grants (No. rfc7523).

37. Paksula, M. (2010). Persisting objects in redis key-value database. University of Helsinki, Department of Computer Science, 27.

38. Ionescu, V. M. (2015, September). The analysis of the performance of RabbitMQ and ActiveMQ. In 2015 14th RoEduNet International Conference-Networking in Education and Research (RoEduNet NER) (pp. 132-137). IEEE.

39. Bernardi, S., Donatelli, S., & Merseguer, J. (2002, July). From UML sequence diagrams and statecharts to analysable petri net models. In Proceedings of the 3rd international workshop on Software and performance (pp. 35-45).

40. Indriana, M., & Adzani, M. L. (2017, November). UI/UX analysis & design for mobile e-commerce application prototype on Gramedia. com. In 2017 4th International Conference on New Media Studies (CONMEDIA) (pp. 170-173). IEEE.

41. Яковлева О. В., Ардасов В.А. (2022) Розробка методу для розпізнавання учасників он-лайн конференцій на основі аналізу відеопотіку, Abstracts of XXXVII International Scientific and Practical Conference «Modern ways of solving the latest problems in science» (September 20 – 23, 2022). Varna, Bulgaria, pp. 426-429.

42. Яковлева О. В., Ардасов В. А. (2021) Розробка та дослідження методу детектування медичних масок на обличчях, С. 151.