Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____
(повна назва)

Кафедра_____Програмної інженерії_____
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти – другий (магістерський)

Дослідження підходів до розробки системи сегментації тіла людини на потоковому відео стосовно пози людини (тема)

Виконав: студент 2 курсу, групи ІПЗм-18-2___
_____Богомаз А.С._____
(прізвище, ініціали)

спеціальності 121- Інженерія програмного забезпечення
(код і повна назва спеціальності)

Освітньо-наукової програми___
(тип програми)

Інженерія програмного забезпечення
(повна назва освітньої програми)

Керівник _____проф. Білоус Н.В._____
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри, проф. _____ З.В.Дудар

2020 р.

Харківський національний університет радіоелектроніки

Факультет_Комп'ютерних наук_____

Кафедра_Програмної інженерії_____

Рівень вищої освіти – другий (магістерський)

Спеціальність _121-Інженерія програмного забезпечення_____
(код і повна назва)

Тип програми освітньо-наукова програма_____

Освітня програма_Інженерія програмного забезпечення_____

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____»_____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові _Богомазу Антону Сергійовичу___
(прізвище, ім'я, по батькові)

1. Тема роботи _Дослідження підходів до розробки системи сегментації тіла людини на потоковому відео стосовно пози людини_____

затверджена наказом університету від "_27_" __03__2020 р № _473Ст_
заповнюється вручну після отримання наказу

2. Термін подання студентом роботи до екзаменаційної комісії
18 червня 2020 р.

3. Вихідні дані до роботи *електронні ресурси за обраною тематикою, вимоги до функціональності програми, методи порівняння поз, платформа розробки NodeJs СУБД MongoDB, середовища розробки VisualStudio Code*

4. Перелік питань, що потрібно опрацювати в роботі *мета роботи, аналіз предметної галузі та існуючих рішень, постановка задачі, дослідження методів визначення поз та методів порівняння поз, дослідження можливості застосування їх для обробки потокового відео*

5 Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спецчастина	проф. Білоус Н. В.		18.05.20

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1.	Аналіз проблемної області	27.01.20 – 10.02.20	виконано
2.	Постановка задачі	11.02.20 – 13.02.20	виконано
3.	Дослідження існуючих методів та моделей	14.02.20 – 19.02.20	виконано
4.	Розробка математичних моделей та алгоритмів	20.02.20 – 3.03.20	виконано
5.	Розробка бази даних та архітектури	4.03.20 – 7.03.20	виконано
6.	Програмна реалізація	8.03.20 – 25.03.20	виконано
7.	Експериментальне дослідження розроблених алгоритмів	26.03.20 – 30.03.20	виконано
8.	Підготовка пояснювальної записки	1.04.20 – 29.04.20	виконано
9.	Підготовка презентації та доповіді	30.04.20 – 8.05.20	виконано
10.	Попередній захист	19.05.20	виконано
11.	Нормоконтроль, рецензування	11.05.20 – 15.05.20	виконано
12.	Занесення диплома в електронний архів	18.05.20	виконано
13.	Допуск до захисту у зав. кафедри	18.05.20	виконано

Дата видачі завдання _____ _____2020 р.

Студент _____ _____Богомаз А. С._____
(підпис)

Керівник роботи _____ _____проф. Білоус Н. В._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Звіт з науково-дослідної практики: 87 с., 22 рис., 4 табл., 17 джерел.

ВИЗНАЧЕННЯ ПОЗ, ПОРІВНЯННЯ ПОЗ, КОМП'ЮТЕРНИЙ ЗІР, POSENET, JAVASCRIPT

Метою роботи є створення порівняльної характеристики бібліотек для визначення пози людини на зображенні та методів порівняння поз в контексті розробки системи для перевірки правильності виконання вправи у реальному часі.

Методами дослідження є емпіричні методи дослідження, а саме: вимірювання якості та швидкості роботи методів та бібліотек, порівняння результатів та експериментальне дослідження отриманих результатів.

В результаті створено порівняльну характеристику існуючих бібліотек для визначення поз та методів порівняння поз. На підставі цих досліджень була побудована система визначення правильності виконання вправ у реальному часі. Було проведено дослідження отриманої системи

POSE ESTIMATION, POSE COMPARISON, COMPUTER VISION, POSENET, JAVASCRIPT

The purpose of this work is to investigate libraries that provide pose estimation functionality and methods of pose comparison in terms of development real-time workout checking system.

Research methods are empirical research methods, namely: measuring the quality and speed of methods and libraries, comparing results and experimental study of the results.

As a result, pose estimation libraries and pose comparison methods were investigated and their comparison characteristic was developed. Based on these investigations, real-time workout checking system was built. Research of developed system was conducted.

ЗМІСТ

ВСТУП

Однією з задач комп'ютерного зору є задача визначення тіла людини на зображенні. Існує багато методів вирішення цієї задачі, деякі базуються на специфічному обладнані (motion capture, kinect) та надають найбільшу точність, деякі дають меншу точність, але не потребують додаткового обладнання та використовують меншу обчислювальну потужність. Останні мають дуже широку сферу застосування, наприклад, для індивідуальних спортивних тренувань або при реабілітаційних вправах. В наведених сферах велику значимість має вирішення задачі порівняння пози людини з еталонною для виявлення правильності виконання тієї чи іншої вправи.

Проблема порівняння поз полягає в тому, що різні люди майже ніколи не будуть мати однакові пропорції тіла, тому одні й ті ж самі пози не будуть мати повного збігу, а це означає, що вони повинні мати деяку градієнтну шкалу порівняння. Однаковими будуть вважатися пози, що перевищують якусь визначену планку. Іншою проблемою є те, що люди стоять на різних відстанях від камери та під різними кутами. Тому необхідно проводити нормалізацію зображення при виконанні порівняння.

Існує багато досліджень різних методів визначення поз на зображенні, але методи порівняння поз дуже мало описані в існуючій літературі. Тому дослідження методів порівняння визначених на зображені поз є актуальними на сьогодні.

На кафедрі програмної інженер проводяться дослідження в області комп’ютерного зору, зокрема підходів до задачі розпізнавання облич, що базуються на вейвлет трансформації [1] та реалізації алгоритму Віоли-Джонса. У дипломній роботі розширено сферу дослідження на розпізнавання пози людини на відео та порівняння поз.

Метою роботи є створення порівняльної характеристики бібліотек для визначення пози людини на зображенні та методів порівняння поз в контексті розробки системи для перевірки правильності виконання вправи у реальному часі.

Об'єктом дослідження є бібліотеки для визначення пози людини на зображенні та методи порівняння поз.

Предмет дослідження: створення порівняльної характеристики бібліотек визначення пози людини на зображенні та методів порівняння, розробка системи для перевірки правильності виконання вправи у реальному часі.

Методами дослідження є емпіричні методи дослідження, а саме: вимірювання якості та швидкості роботи методів та бібліотек, порівняння результатів та експериментальне дослідження отриманих результатів.

В ході роботи було проведено порівняльний аналіз існуючих бібліотек визначення поз з відкритим початковим кодом. Серед них було обрану ту, що найбільш відповідає меті роботи, та проведено розширене дослідження її роботи, а саме дослідження точності та швидкості визначення поз при різних конфігураціях бібліотеки.

Для вивчення методів порівняння поз було зібрано датасет зображень, що складається з 521 зображення. Ці зображення об’єднані в групи, кожна з яких включає від 7 до 13 зображень людей в однакових позах. Було використано зображення кінцевих позицій при виконанні фізичних вправ.

На підставі цього набору даних та результатів визначення поз бібліотекою PoseNet вперше було створено порівняльну характеристику існуючих методів порівняння поз. Результати дослідження дозволяють визначити найбільш точний метод порівняння поз, а також визначити можливості використання сучасних бібліотек визначення поз для обробки відео у реальному часі для вирішення прикладних задач.

Для підтвердження теоретичних досліджень та зроблених рекомендацій було розроблено систему визначення правильності виконання вправ користувачем у режимі реального часу.

За результатами дослідження була написана та подана до опублікування до наукового журналу «Радіоелектроніка, інформатика, управління» («РІУ») стаття «Дослідження методів визначення та порівняння поз на потоковому відео» (див. додаток В).

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Однією з задач комп’ютерного зору є визначення пози тіла людини на зображенні. Вирішення цієї проблеми має застосування у багатьох сферах людської діяльності, від медицини до криміналістики. Наприклад, система, що за позою або ходою людини визначала би, чи є в неї проблеми з осанкою чи опорно-руховим апаратом, та могла би зберігати та обробляти дані про хід реабілітації.

Задачу визначення пози людини можна визначити як задачу пошуку точок сполучення на тілі людини, також відомих як ключові точки – лікті, зап’ястя, коліна та інші. Знайшовши ці точки можна, побудувати скелет людини й таким чином визначити безпосередньо позу людини. Це і є сегментація тіла людини на зображенні.

Розглянемо класичні методи вирішення цієї задачі. Можна розглянути скелет людини як граф у якому вершинами є точки сполучення, а кістки – ребрами. Надалі необхідно розробити математичну модель що буде визначати вірогідність знаходження вершин графа у тій чи іншій точці зображення враховуючи реалістичність такого місцезнаходження. Найвідомішою з таких моделей є pictorial structures model [2]. Іншим класичним варіантом вирішення цієї задачі є пошук із заздалегідь розмічених зображень найбільш близького до наданого і використати його розмітку. Звичайно цей засіб є найбільш неточним але найшвидшим і найпростішим у реалізації і для деяких випадків може бути цілком достатнім.

Сучасними методами пошуку є методи що базуються на згорткових нейронних мережах. «DeepPose» [3] була першою великою доповіддю, яка застосувала глибоке навчання до визначення пози людини. Новий підхід досяг результатів роботи SOTA і переміг існуючі моделі. У цьому підході оцінка пози формулюється як проблема регресії на основі згорткових нейронних мереж до суглобів тіла. Вони також використовують каскад таких регресорів для уточнення оцінок пози і отримання кращих оцінок. Найважливіше нововведення цього підходу це те, що він розглядає позу в цілому, тобто навіть якщо певні суглоби

приховані, їх можна визначити, якщо поза обгрунтована цілісно. У роботі стверджується, що згорткові нейронні мережі, істотно, надають такого роду обгрунтування та демонструють сильні результати.

Задача сегментації тіла людини може полягати не лише у двовимірному просторі, а також у трьохвимірному. У трьохвимірному просторі повинна будуватись об'ємна модель тіла людини. Є декілька існуючих підходів для побудови тривимірної моделі тіла людини. Найточнішим на сьогодні є підхід з використанням спеціальних датчиків на тілі або одязі людини, за якими відбувається побудова ключових точок на 3D-моделі. Такі технології звуться Motion Capture та активно використовуються у кінематографі при побудові спецефектів та мультиплікації. Однак їх неможливо запровадити та використовувати поза умов спеціально побудованої студії.

Для повсякденного використання є необхідність у реалізації системи, що дозволяє отримувати 3D-модель з використанням однієї камери та аналізу зображення. Основними складнощами у порівнянні з двовимірною сегментацією тіла є більший простір для поз у трьох вимірах ніж у двох, та наявність великої кількості невизначеностей – поз які у проекції виглядають однаково, а у тривимірному просторі є різними. Також складності надає відсутність можливості створення якісних датасетів в умовах реального світу. Усі датасети створюються за допомогою технологій Motion Capture і тому можуть бути створені лише в умовах приміщення. Відсутність якісних датасетів є дуже вузьким місцем у розвитку цієї технології, тому що саме на них базується навчання нейронної мережі.

Серед існуючих підходів до сегментації тіла людини у тривимірному просторі варто виділити наступні: використання pictorial structures model – моделі, що використовується у 2D сегментації, дискримінаційні методи та Deep Learning [4].

Звичайний pictorial structures model розглядає тіло людини як суглобову структуру. Модель, як правило, складається з двох термінів, які моделюють зовнішній вигляд кожної частини тіла та просторове співвідношення між сусідніми

частинами. Оскільки довжина кінцівки в 2D може змінюватись, для моделювання кожної частини тіла була запропонована суміш моделей.

Дискримінаційні методи розглядають сегментацію як проблему регресії. Після вилучення функцій із зображення, співставлення отримується з простору зображень до простору пози. Через шарнірну структуру скелета людини місця розташування суглобів сильно корелюються. Щоб розглянути залежності між вихідними змінними, пропонується використовувати структурований SVM для вивчення співставлення функцій сегментації та спільних локацій.

Deep Learning підхід полягає у тому, шо замість розгляду структурних залежностей вручну, більш прямим способом є «вбудовування» структури у функцію відображення та вивчення подання, яке роз'єднує залежності між вихідними змінними [5]. У цьому випадку моделі потребують виявлення шаблонів людської пози з даних, що зазвичай вимагає великого набору даних для навчання.

Першим кроком до вирішення задачі сегментації тіла є пошук безпосередньо людини на зображенні. Потрібно мати на увазі, що людина може бути не одна і це може ускладнити побудування скелету. Зображення можна поділити на 3 категорії: зображення однієї людини, зображення декількох людей, що не перетинаються одне з одним та зображення людей, що перетинаються на зображенні (див. рис. 1.1).

Рис. 1.1 – Різні розташування людей відносно одне одного на зображені

Для першого типу зображень пошук людини на зображень потрібен для зменшення області пошуку сполучень. Для другого – для зменшення області

пошуку та для того, щоб на етапі поєднання точок сполучення поєднувати між собою точки, що належать одній й тій самій людині. Найскладнішим випадком є третій тип зображень, коли люди перетинаються на зображенні і потрібно розподіляти їх між собою. При цьому, пошук точок сполучення може виконуватись як після відокремлення людей, так і до нього, в залежності від алгоритму. Існують алгоритми, що розрізняють зображення людей на підставі знайдених точок сполучення. Вони працюють також і для сегментації тіла на зображеннях з однією людиною та мають меншу швидкість у порівнянні з алгоритмами, що заздалегідь опрацьовують одну людину, однак мають кращий показник точності.

1.1 Метрики ефективності моделей комп'ютерного зору

У завданнях машинного навчання для оцінки якості моделей і порівняння різних алгоритмів використовуються метрики, їх вибір та аналіз – важлива частина будь-якого дослідження. В задачах визначення об'єктів найрозповсюдженішою метрикою є mAP (Mean Average precision) – показник середньої точності. Вона не є єдиною, але саме його більш за все наводять у сучасних доповідях щодо алгоритмів сегментації. Зокрема нього також існують R-CNN, SSD тощо.

IoU (Intersection over union) – відношення перетину двох областей до їх об'єднання. Використовується для визначення, наскільки правильно алгоритм виділив об'єкт відносно його реального місцезнаходження.

$$IoU = \frac{S_{overlap}}{S_{union}}, \tag{1.1}$$

де $S_{overlap}$ – площа перетину двох областей;

S_{union} – їхня загальна площа.

На рисунку 1.2 подано приклад результату роботи алгоритму сегментації. Синім виділено реальну область, де розташовано об’єкт, а червоним – область, яку визначив алгоритм. У деяких наборах даних заздалегідь визначається поріг IoU (скажімо, 0,5), класифікуючи, чи є прогноз справді позитивним чи хибним позитивним.

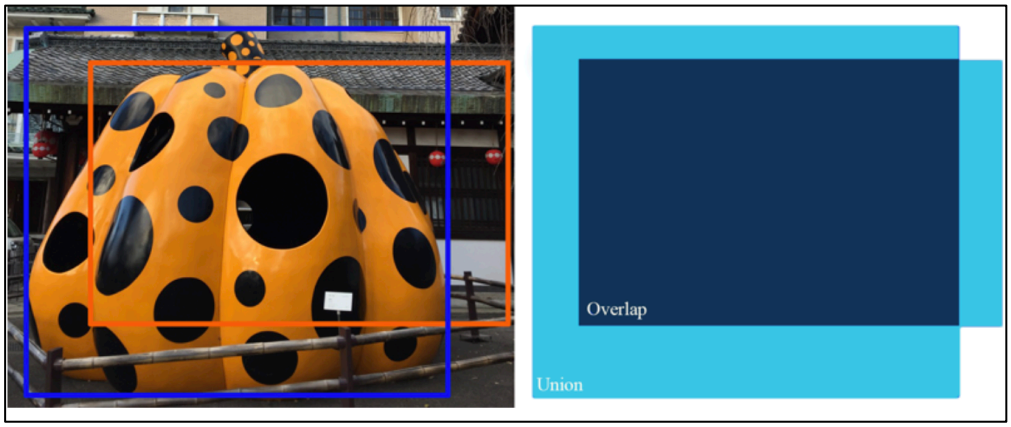

Рис. 1.2 – Приклад сегментації об’єкта

Матриця помилок (confusion matrix) – термін, що використовується для визначення типів помилок класифікації. Припустимо, що у нас є два класи і алгоритм, який визначає приналежність кожного об'єкта одному з класів, тоді матриця помилок класифікації буде виглядати так, як показано на рисунку 1.3.

	$y = 1$	$y = 0$
$\hat{y} = 1$	True Positive (TP)	False Positive (FP)
$\hat{y} = 0$	False Negative (FN)	True Negative (TN)

Рис. 1.3 – Матриця помилок класифікації

Тут ý – це відповідь алгоритму на об'єкті, а y – справжня мітка класу на цьому об'єкті. Таким чином, помилки класифікації бувають двох видів: False Negative (FN) і False Positive (FP).

Найочевиднішою і найпростішою метрикою для розуміння є accuracy, що являє собою долю правильних відповідей алгоритму і може бути аналітично записана наступним чином:

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN}, \tag{1.2}$$

Ця метрика не є репрезентативною для задач класифікації з нерівними класами. Її результат є загальним значенням і якщо алгоритм правильно розрізняє один клас елементів, що виражений більшістю елементів з усієї вибірки, величина буде відповідати великій якості класифікації, що не є реальним станом речей. Тому значення метрики повинно якнайменше залежати від тестових даних.

Цю проблему вирішують метрики precision (точність) та recall (повнота). Вони оцінюють якість роботи алгоритму на кожному з класів окремо.

$$precision = \frac{TP}{TP + FP}, \tag{1.3}$$

$$recall = \frac{TP}{TP + FN}. \tag{1.4}$$

Precision можна інтерпретувати як частку об'єктів, названих класифікатором позитивними і при цьому дійсно є позитивними, а recall показує, яку частку об'єктів позитивного класу з усіх об'єктів позитивного класу знайшов алгоритм. Саме введення precision не дозволяє нам записувати всі об'єкти в один клас, так як в цьому випадку ми отримуємо зростання рівня False Positive. Recall демонструє здатність алгоритму виявляти даний клас взагалі, а precision – здатність відрізняти цей клас від інших класів.

Ці дві метрики можуть використовуватися як окремо так і при розрахунку комплексних метрик, одною з яких є F1-score, що являє собою середнє гармонійне між значеннями precision та recall.

Для подальшого розгляду поняття Average Precision [6] наведемо в якості прикладу спрощену модель дослідження роботи алгоритму визначення об’єкта. Припустимо, що ми маємо набір зображень, що складається з 10 зображень, 5 з яких – зображення котів. Також ми маємо програму, що реалізовує метод який визначає чи є кіт за зображені, чи ні. Результати аналізу зображень цією програмою наведено на рисунку 1.4.

Rank	Correct?	Precision	Recall
1	True	1.0	0.2
2	True	1.0	0.4
3	False	0.67	0.4
4	False	0.5	0.4
5	False	0.4	0.4
6	True	0.5	0.6
7	True	0.57	0.8
8	False	0.5	0.8
9	False	0.44	0.8
10	True	0.5	1.0

Рис. 1.4 – Результати роботи алгоритму класифікації

Побудувавши графік залежності точності від повноти отримуємо зигзагоподібну криву. Загальне визначення середньої точності (AP) – це знаходження області під кривою точност-повноти. Точність і повнота завжди знаходяться між 0 і 1. Отже, AP також потрапляє в межах 0 і 1. Перш ніж обчислити AP для виявлення об'єкта, ми часто спочатку згладжуємо зигзагоподібний графік (див. рис. 1.5).

Графічно для кожного значення recall ми замінюємо кожне значення precision на максимальне значення precision праворуч від цього значення recall.

Математично ми замінюємо значення precision для recall r̂ максимальним значенням precision для будь-якого recall ≥ r̂.

Для розрахунків використовують інтерпольоване значення AP. Це означає, що ось recall розбивають на рівнозначні частини та обчислюють середнє між відповідними значеннями precision. У Pascal VOC2008 [7] обчислюється середнє значення для 11-бальної інтерпольованої AP. Наміром таким чином інтерполювати криву recall-precision є необхідність зменшити вплив «хитання» на криву recall-precision, спричинене малими варіаціями ранжирування прикладів.

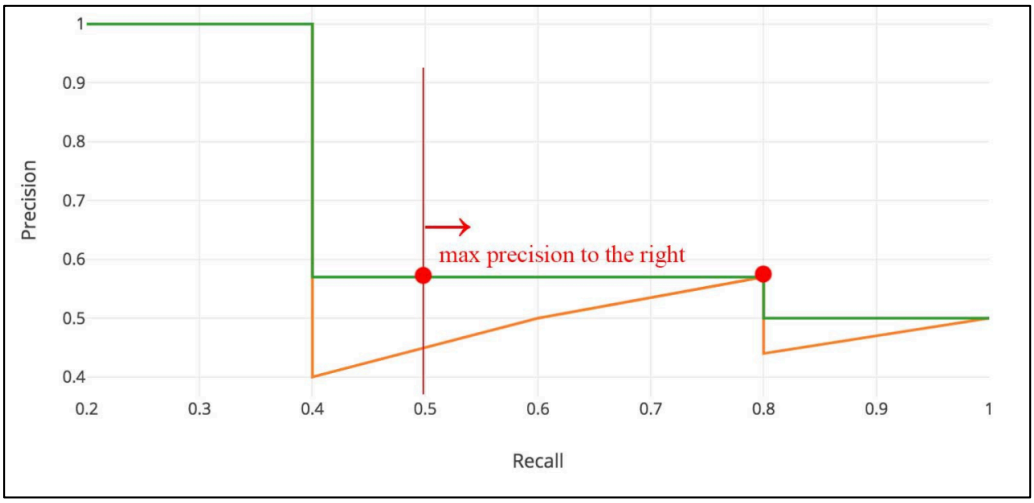

Рис. 1.5 – Графік recall-precision та його згладжена версія

Сучасні наукові доповіді, як правило, описують результати лише для датасету COCO [8], в якому використовується інтерполяція за 101 точкою. В ньому AP – це середній показник для декількох IoU (мінімальний IoU, щоб розглянути позитивну відповідність). AP@[.5:.95] відповідає середньому AP для IoU від 0,5 до 0,95 з розміром кроку 0,05. Також AP в COCO датасеті є середнім значенням для класифікації 80 різних категорій об'єктів (якщо не наведено іншого). Традиційно такий підхід зветься mAP (mean average precision).

Для тестування визначення ключових точок на тілі людини та їх з'єднань замість IoU використовують показник OKS (Object Keypoint Similarity) – схожіть ключових точок об'єкта. Він обчислюється з відстані між прогнозованими точками та розміченими точками, нормалізованими за масштабом людини. Слід зауважити

що цей підхід може бути використано лише для порівняння поз на однакових зображеннях, так як цей метод було створено для саме такого використання і він не враховує різницю у пропорціях тіла різних людей.

Константа масштабу та ключової точки, необхідна для вирівнювання важливості кожної ключової точки: розташування шиї точніше, ніж розташування стегна.

$$OKS = \exp\left(-\frac{d_i^2}{2s^2k_i^2}\right), \tag{1.5}$$

де di – евклідова відстань між реальною ключовою точкою та передбачуваною ключовою точкою;

s – масштаб: площа обмежувального поля, поділена на загальну площу зображення;

k – константа що визначена окремо для кожної контрольної точки.

Константи для ключових точок обчислені групою дослідників з MS COCO.

1.2 Датасет MS COCO

Для розробки нейронних мереж одним з найважливіших елементів є набір даних за яким мережа навчається, тестується та валідується. В залежності від класу та напрямку діяльності нейронної мережі використовуються різні типи даних. Основними характеристиками датасетів є розмір, наявність усіх можливих класів даних в близький до рівних пропорціях, реалістичність даних та якість опису даних.

Для задач сегментації об’єктів на зображенні найпопулярнішим датасетом є COCO dataset. Він містить понад 330 тис. зображень з яких понад 200 тисяч анотовані. Однією з особливостей цього датасету є впроваджений формат анотацій.

Набір даних COCO представлено об’єктом в форматі JSON, що являє собою набір значень «info», «licenses», «images», «annotations», «categories» (в більшості випадків).

Розділ «інформація» містить інформацію високого рівня про набір даних. Якщо ви створюєте свій власний набір даних, ви можете заповнити опис, рік створення, силку на набір даних, дату створення та автора.

Розділ «ліцензії» містить перелік ліцензій, які застосовуються до зображень у наборі даних. Якщо ви ділитесь чи продаєте свій набір даних, ви повинні переконатися, що ваші ліцензії вказані правильно та що ви не порушуєте авторські права.

Розділ «зображення» містить повний перелік зображень у вашій наборі даних. У цій частині немає міток, рамок або сегментів, це просто список зображень та інформації про кожне.

Розділ «аннотації» містить колекцію анотацій. COCO має п'ять типів анотацій: для виявлення об'єктів, виявлення ключових точок, сегментації матеріалів, паноптичної сегментації та підпису зображень. Анотації зберігаються у форматі JSON. Він містить список кожної окремої примітки про об'єкт із кожного зображення в наборі даних. Наприклад, якщо на 100 зображень розміщено 64 велосипеди, буде 64 велосипедні анотації (разом з тоннами анотацій для інших категорій об'єктів). Часто на зображенні буде кілька об’єктів одного типу. Зазвичай це призводить до появи нового анотаційного елемента для кожного.

Об'єкт «категорії» містить перелік категорій (наприклад, собака, човен), і кожна з них належить до суперкатегорії (наприклад, тварина, транспортний засіб). Оригінальний набір даних COCO містить 90 категорій. При створенні нового датасету або при доповненні існуючого, можуть бути використані існуючі категорії COCO або створений абсолютно новий власний список.

При вирішенні задач сегментації частин тіла, або для задач пошуку ключових точок, використовують частину датасету, що має відповідні анотації. Станом на версію 2017 року, у наборі даних COCO є лише одна категорія «людина» з

ключовими точками, але теоретично це може бути розширено до будь-якої категорії, яка може мати різні цільові точки.

Що стосується людини, «ключові точки» вказують різні частини тіла. «Скелет» позначає з'єднання між точками. Ця інформація є загальною для всієї категорії, тобто вона буде однаковою для всіх зображень людей незалежно від того, які елементи можуть бути визначені на зображені, а які – ні. На рисунку 1.6 наведено приклад опису категорії людини. В описі скелета [6, 8] означає, що «left_shoulder» поєднується з «left_elbow».

```
"categories": [
    {
        "supercategory": "person",
        "id": 1,
        "name": "person",
        "keypoints": [
            "nose","left_eye","right_eye","left_ear","right_ear",
            "left_shoulder","right_shoulder","left_elbow","right_elbow",
            "left_wrist","right_wrist","left_hip","right_hip",
            "left_knee","right_knee","left_ankle","right_ankle"
        ],
        "skeleton": [
            [16,14],[14,12],[17,15],[15,13],[12,13],[6,12],[7,13],[6,7],
            [6,8],[7,9],[8,10],[9,11],[2,3],[1,2],[1,3],[2,4],[3,5],[4,6],[5,7]
        ]
    }
]
```

Рис. 1.6 – приклад опису категорії у датасеті COCO

Конкретне положення ключових точок на зображенні позначено в розділі «annotations» (див. рис. 1.7). Розділ являє собою масив значень, що описують подані у розділі «categories» ключові точки. Для кожної точки в масиві міститься 3 значення: координати x та y на зображенні та v, що може приймати значення від 0 до 2. Значення 0 є позначкою, що ключова точка не відмічена, в такому випадку x та y координати також дорівнюють 0. Значення 1 відповідає ситуації, коли точка

позначена, але на зображенні її не видно, наприклад якщо зрозуміло де вона знаходиться, але на зображенні вона закрита іншим об’єктом. Значення 2 означає, що точка видна та описана. Це використовується при аналізі результатів визначення пози програмою, бо саме такі ключові точки здебільшого створюють помилки при визначенні пози. Ця інформація надає змогу зрозуміти що саме було не так при виконанні визначення.


```
"annotations": [
    {
        "segmentation": [[204.01,306.23,...206.53,307.95]],
        "num_keypoints": 15,
        "area": 5463.6864,
        "iscrowd": 0,
        "keypoints": [229,256,2,...,223,369,2],
        "image_id": 289343,
        "bbox": [204.01,235.08,60.84,177.36],
        "category_id": 1,
        "id": 201376
    }
]
```


Рис. 1.7 – Приклад анотації для сегментації тіла людини

Зокрема опису ключових точок анотації містять інформацію про сегментацію фігури людини на зображенні, кількість ключових точок, позначених на зображенні, площу у пікселях, яку займає фігура людини на зображенні, ідентифікатор зображення та рамку, що оточує фігуру людини. Порівняно з загальним набором даних створеним для сегментації об’єктів на зображенні, специфічними даними для визначення пози людини є лише кількість ключових точок та значення їх розташування.

Також варта уваги позначка «iscrowd». Коли на зображенні чи на фрагменті можна чітко виявити один елемент відповідної категорії вона буде мати значення 0 і анотація повністю відповідає опису цього елемента. Коли на зображенні чи його фрагменті є декілька елементів одного классу, що сильно перетинаються та не можуть бути відокремлені, ця позначка має значення 1. Коли виділяється конкретний елемент, розділ «segmentation» містить масив точок, що позначають замкнену ламану, що відповідає границям виділеного об’єкта. В випадку коли «iscrowd» дорівнює 1, сегментація описується показниками «size» та «counts». Size описує розміри фрагменту зображення в пікселях, а «counts» масив кількості пікселів, що йдуть послідовно на зображенні та в залежності від порядку в масиві мають значення «елемент присутній на цих пікселях» та «елемент відсутній на цих пікселях». Тобто, якщо масив виглядає наступним чином: [30, 42, 56, 78], це позначає, що перші 30 пікселів не містять об’єктів сегментації, наступні 42 містять, наступні 56 не містять на останні 78 містять.

1.3 Огляд методів порівняння поз

В результаті роботи моделі ми отримуємо опис ключових точок з зображення, а саме інформацію про тип ключової точки (ліве плече, правий лікоть і т.п.), її координати на зображенні та коефіцієнт точності її визначення. Для порівняння використовують векторні величини. Тож результати визначення поз з двох зображень необхідно привести до подібних векторів.

При порівнянні результатів визначення ключових точок на зображенні та його фактичній розмітці використовується показник OKS, що при розрахунку використовує значення масштабу, що рахується як відношення площі рамки навколо людини, до площі всього зображення. Цей метод не може бути використаний при порівнянні поз різних людей на різних зображеннях.

Проблемою є те, що зображення можуть мати різний початковий розмір та різне положення людей у кадрі, що впливає на результати порівняння. Для вирішення цієї проблеми обрізаємо зображення залишаючи лише зображення людини. Наступним кроком для приведення векторів до тотожного виду потрібно нормалізувати вектори за допомогою L2 нормалізації.

При порівнянні поз нам потрібно визначити ступінь схожості векторів, бо вони ніколи не будуть 100% ідентичними. Для такого визначення використовують поняття відстані між векторами. Найпростішим та класичним способом її визначення є евклідова відстань, що рахується за формулою:

$$d = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}, \tag{1.6}$$

де d – евклідова відстань,

n – розмірність векторів,

x,y – відповідні координати двох векторів в площині вимірювання і.

Але через нормалізацію векторів цей метод в чистому вигляді втрачає свою репрезентативність, бо зменшення розміру зображення прямо впливає на результати його обчислення. тому можна використати поняття косинусоїдної схожості, що є значенням косинусу кута між векторами, що вираховується за формулою:

$$\cos(\mathrm{a}, \mathrm{b}) = \frac{a * b}{|a| * |b|}. \tag{1.7}$$

Використовуючи значення косинусоїдної схожості, можемо розрахувати значення відстані між векторами за формулою:

$$D\left(F_{xy}, G_{xy}\right) = \sqrt{2 * (1 - cosineSimilarity(F_{xy}, G_{xy}))}. \tag{1.8}$$

В результаті отримуємо значення, завдяки якому можна оцінити схожість поз. Чим менше це значення, тим більше схожі пози. Отримана схожість

коливається від −1, що означає точно протилежне, до 1 означає абсолютно однакове.

Іншим методом є метод з урахуванням вірогідності правильного знаходження контрольної точки. Така вірогідність може буди надана бібліотекою розпізнавання і вона вказує на рівень «впевненості» в тому, що суглоб знаходиться саме у визначеній точці, а не у деякій іншій. Іноді ми точно знаємо, де знаходиться суглоб, наприклад, якщо ми можемо це чітко бачити; в інших випадках у нас дуже низька впевненість, наприклад, якщо суглоб відрізаний або закритий. Попередня фільтрація зображення [9] може покращити ці значення, але лише в окремих випадках. Якщо ми ігноруємо ці показники достовірності, ми втрачаємо цінні відомості про свої дані.

Для використання цієї інформації дослідники з корпорації Google Джордж Папандреу та Тайлер Чжу розробили формулу [10] :

$$D(F,G) = \frac{1}{\sum_{k=1}^{n} F_{c_k}} * \sum_{k=1}^{n} F_{c_k} * |F_{xy_k} - G_{xy_k}|, \quad (1.9)$$

де G і F – два вектори поз, що порівнюються після L2 нормалізації;

n – кількість визначених контрольних точок;

F_{ck} – значення ймовірності правильного знаходження суглоба для елемента за номером k вектору F;

F_{xy} і G_{xy} – x і y позиції k-ої ключової точки для кожного вектору.

Третій метод [11] порівняння не потребує попередньої нормалізації координат, але вектор будується за іншим принципом. Для кожних трьох анатомічно поєднаних між собою точок вираховується косинус кута, між отриманими частинами тіла за формулою (1.8). Відмінністю від косинусоїдної схожості, що використовується у першому методі, є те, що в даному випадку отримується значення для двовимірного вектору, що описує положення двох кінцівок відносно одне одного.

1.4 Аналіз існуючих бібліотек для визначення пози на зображенні

В результаті дослідження існуючих бібліотек для визначення пози, було виявлено, що велика кількість з них – пропрієтарні, тож можуть бути використаниі лише компаніями-розробниками або за ліцензією від розробника. Тож для ознайомлення були обрані ті, що мають відкритий початковий код, а саме: HRNet, PoseNet та OpenPose.

HRNet містить під собою метод визначення пози людини «знизу вгору». Це означає, що метод спочатку аналізує зображення у повному розмірі і потім обробляє більш детального його менші частини [12]. Особливістю реалізації є те, що вона дозволяє вирішити проблему зміни масштабу в оцінці пози для багатолюдних зображень методами «знизу вгору» та більш точно локалізувати ключові точки, особливо для зображень людей у маленькому масштабі. Ця модель може бути інтегрована в програмну систему за допомоги мови програмування python з використанням бібліотеки OpenCV.

OpenPose має API для python та plugin для ігрового двигуна Unity. Всередині він використовує мультіпоточну оптимізацію, що дозволяє прискорити швидкість обробки зображення та відповідно знаходити більше контрольних точок на зображенні в умовах потокового відео. Згідно офіційної документації OpenPose може визначати 25 ключових точок при оцінці тіла/ніг, 2x21 ключову точку при оцінці рук та 70 точок при аналізі зображення обличчя [13].

Бібліотека PoseNet базується на фреймворку TensorFlow Light та вміє розрізняти 17 ключових точок на зображені. Важливою деталлю, яку слід зазначити, є те, що дослідники підготували як ResNet, так і MobileNet модель PoseNet. Модель ResNet має більш високу точність, але має великий розмір і багато шарів, у той час як модель MobileNet розроблена для роботи на мобільних пристроях [14]. Бібліотека може бути використана за допомогою великої кількості мов програмування, а саме в Python, C++, Java, Swift, Objective C та Javascript.

Порівняльні результати тестування якості цих бібліотек на датасеті MS COCO наведено в таблиці 1.1.

Таблиця 1.1 – Результати порівняння точності сегментації тіла людини

	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}
PoseNet	69.6	86.3	76.6	65.0	76.3
OpenPose	70.4	88.6	77.8	67.0	76.9
HRNet	70.0	87.8	76.1	64.8	77.3

В таблиці наведені різні показники AP (Average Precision): загальне середнє значення (mAP), значення для OKS рівному 0,50 та 0,75, а також окремо для відокремлення визначення об’єктів великого та середнього розміру. Об’єктами середнього розміру вважаються такі, чия розміри знаходяться поміж 32x32 та 96x96 пікселів. Об’єкти меншого розміру не мають відповідної розмітки.

Для реалізації обробки потокового відео мовою Python, яку підтримують всі з означених вище бібліотек, є необхідність в реалізації саме клієнт-серверної архітектури, що зумовлено відсутністю підтримки цієї платформи операційними системами iOS та Android, що є найпоширенішими операційними системами для мобільних пристроїв.

Проте аналіз відео-потоку на сервері автоматично призводить до великої затримки між діями користувача та відображенням результатів. Існування браузерної версії бібліотеки PoseNet вирішує цю проблему і надає змогу реалізувати крос-платформний додаток для визначення пози у реальному часі.

2 ПОСТАНОВКА ЗАДАЧІ

2.1 Постановка дослідницької задачі

Для реалізації додатку, що аналізує правильність виконання фізичних вправ необхідно дослідити методи порівняння поз на зображення з відео з еталонними позами. Саме завдяки цьому порівнянню буде визначено чи правильну позу прийняв користувач.

Необхідно дослідити ефективність описаних в попередньому розділі методів порівняння поз при використанні їх для порівняння поз отриманих бібліотекою PoseNet. Ефективність полягає в точності порівняння та в швидкості виконання цього порівняння. Але швидкість порівняння буде приблизно однаковою, так як всі вони розраховуються за формулами і мають обчислювальну складність O(C), тому необхідно дослідити лише точність порівняння.

Дослідження повинно виконуватись у декілька етапів. Перший – створення датасету зображень поз, а також даних про те, які з цих зображень повинні бути визначені як такі, що містять однакові пози, а які – ні. На рисунку 2.1 зображено приклади зображень для датасету.

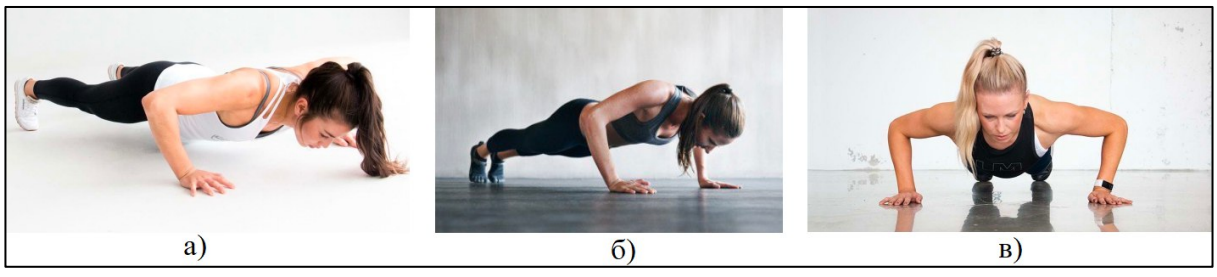

Рис 2.1 – Приклади зображень для датасету

Так як бібліотека аналізує двовимірне зображення і будує скелет у двовимірному просторі, пози будуть визначені як однакові тільки в тому випадку, якщо вони знаходяться на фотографії в одному ракурсі. Зображення а та б мають бути анотовані як такі, що містять однакові пози, у той час як зображення в, хоч і

містить ту ж саму позу, зображує її в іншому ракурсі, тому не може бути анотовано як відповідне до а та б.

Для виконання цього етапу повинен бути реалізований скрипт мовою Javascript, що визначає пози з зображень, використовуючи бібліотеку PoseNet та зберігає отримані дані в базі даних MongoDB. Ця база даних обрана виключно з міркувань зручності використання для збереження нереляційних даних.

Наступним етапом є реалізація описаних у першому розділі методів порівняння та отримання результатів порівняння. Кожним з запропонованих методів порівнюються між собою усі пози визначенні однією конфігурацією бібліотеки. Тобто, якщо в нас є 50 зображень та 3 конфігурації бібліотеки, ми маємо 150 визначених поз, по 50 для кожної конфігурації. Кожним з трьох методів порівняння ми порівнюємо між собою 50 поз, що визначені однією конфігурацією. В результаті кожного порівняння повинно бути отримано запис у базі даних про порівнювані пози, метод визначення поз, метод порівняння поз та результат, що являється значенням відстані між двома позами і приймає значення від 0 до 1, де чим менше отримане значення тим більше пози схожі.

Останнім етапом є розрахування F1-показника. Отримані результати з попереднього етапу групуються за значеннями конфігурації визначення пози та методу порівняння. Пози вважаються однаковими, якщо значення отримане при порівнянні менше за порогове значення. В якості порогових беруться значення від 0.05 до 0.40 з кроком в 0.05. Для кожного порогового значення та для кожної групи результатів порівнянь розраховується F1-показник.

Таким чином, для кожної конфігурації методу визначення пози та для кожного методу порівняння поз, буде отримано 8 значень, що відображають середню правильність визначення чи є пози однаковими, чи ні для 8 порогових значень схожості.

За отриманими результатами можна буде зробити висновки стосовно того, який з наведених методів порівняння поз надає більшу точність. Також можна оцінити яка з конфігурацій бібліотеки PoseNet надає більшу якість визначення пози для порівняння, а також визначити швидкість її роботи при різних конфігураціях.

Очікуваним результатом є рекомендація щодо конфігурації бібліотеки PoseNet та методу порівняння поз, що будуть оптимальними контексті реалізації програми для порівняння поз на потоковому відео у реальному часі.

2.2 Формування вимог до експериментального додатку

З ціллю експериментального підтвердження або спростування результатів дослідження необхідно побудувати систему, що виконує перевірку правильності виконання вправ у реальному часі. Вона повинна використовувати рекомендовані параметри конфігурацій бібліотеки визначення поз та рекомендований метод порівняння поз. Базовий функціонал системи складає авторизація та реєстрація, створення вправ, пошук та фільтрація вправ та модуль перевірки правильності виконання вибраної вправи.

Система повинна складатися з веб-додатку, що підтримується браузерами систем Mac OS, Windows, Linux, iOS та Android, а також з серверною частини, що надає доступ до сховища даних.

Сервер зберігає інформацію про користувачів, вправи, та зберігає файли зображень, з яких складаються вправи. Дані про користувачів це унікальне ім’я у системі і пароль. Пароль має бути збереженим у захищеному вигляді. Вправа складається з назви, автора (ідентифікатору користувача) та колекції зображень в заданій послідовності, що відображають кінцеві пози при виконанні вправи. Саме з цими позами і буде виконуватись порівняння відео користувача.

Система повинна мати дві ролі: «Гість» – користувач, що не має облікового запису у системі та «Користувач» – зареєстрований та авторизований користувач. Можливість створення вправ повинен мати лише зареєстрований користувач, а можливість пошуку і виконання вправ повинні мати обидві ролі.

Процес виконання вправи повинен виглядати наступним чином: користувач бачить зображення з підключеної камери, поверх якого малюється поза, визначена

за допомоги алгоритму визначення поз (див. рис. 2.2). Також на екрані користувач повинен бачити зображення поз, які він повинен прийняти у правильній послідовності. В кожен момент часу поза з відео порівнюється з першою з «не пройдених» поз зі списку. Спочатку всі пози вважаються «не пройденими». В момент, коли поза на відео співпадає з поточною, поза помічається як «пройдена», користувач повинен прийняти позу, що зображена на наступному зображенні. Вправа вважається повністю виконаною, коли всі пози помічені «пройденими».

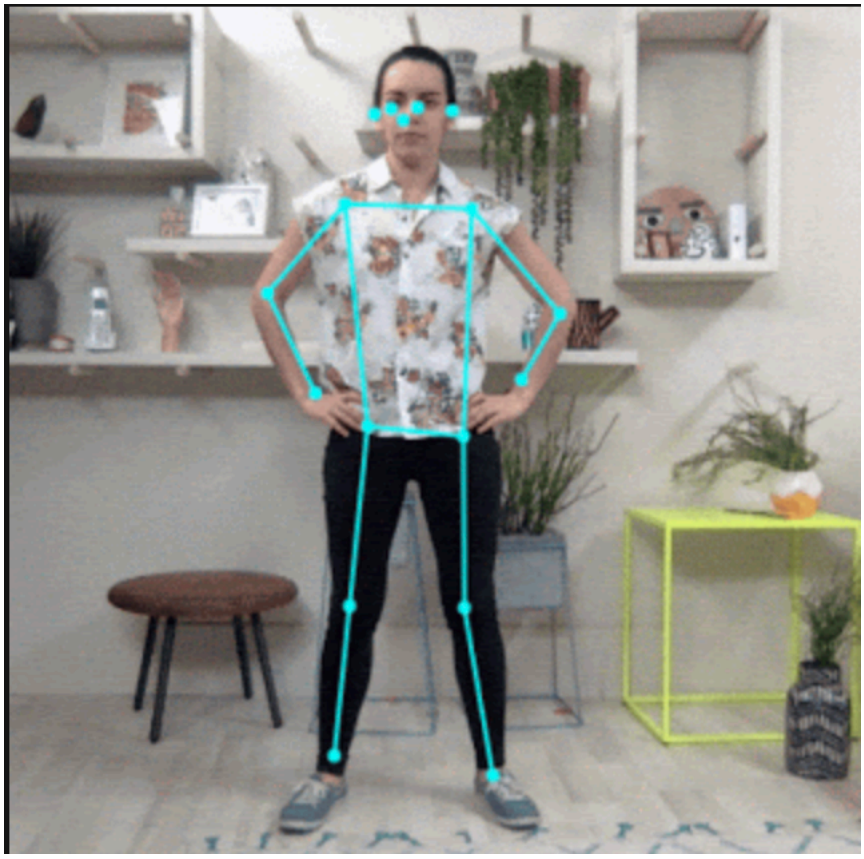

Рис 2.2 – Приклад зображення з веб-камери з доданою схемою пози

Після реалізації необхідно проаналізувати систему з точки зору швидкості визначення пози, надати оцінку працездатності та рекомендації щодо подальшого покращення результатів роботи та розвитку системи. Швидкість визначення визначається у кількості кадрів на секунду, на яких було визначено позу. Цей показник повинен бути максимально наближеним до кількості кадрів на секунду на початковому відео з камери користувача.

3 ОПИС ТЕОРЕТИЧНИХ ТА ПРАКТИЧНИХ ДОСЛІДЖЕНЬ

В ході дослідження було зібрано датасет, що складається з 521 зображення, що об’єднуються в 43 групи поз. Кожна з груп містить зображення одних й тих самих поз з одного ракурсу. Всі пози є зображеннями кінцевих поз при виконанні спортивних вправ (див. рис. 3.1). Відмінності полягають у розмірі зображень, розташуванням людини у кадрі та середовищі у якому знаходиться людина.

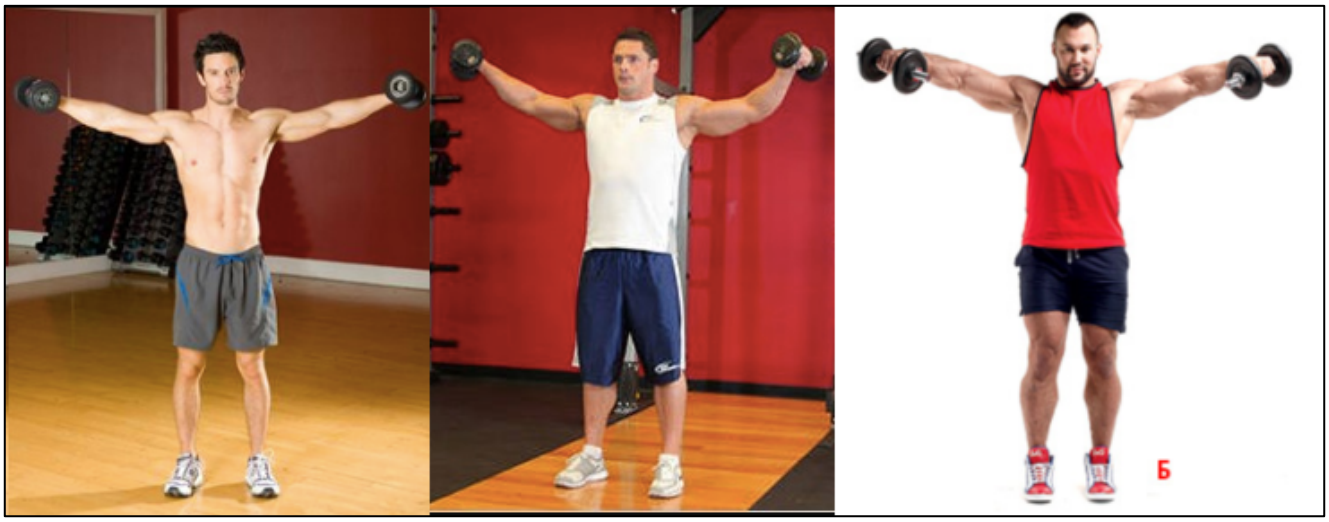

Рис. 3.1 – Приклад зображень з однієї групи в датасеті

PoseNet має можливість гнучкої конфігурації моделі, що впливає на швидкість та точність її роботи. Вона також включає в себе 2 різних архітектури нейронної мережі: «MobileNetV1» та «ResNet50», перша з яких має значно менший розмір та більшу швидкість виконання аналізу зображень, інша надає більшу точність. При аналізі методів порівняння було використано різні конфігурації мережі для отримання опису пози з зображення.

Зокрема архітектури мережі, модель має наступні налаштування. OutputStride – може мати значення 8, 16 або 32 (значення 16 та 32 підтримуються архітектурою ResNet, а повний перелік підтримуються архітектурою MobileNetV1). Він визначає вихідний крок моделі PoseNet. Чим менше значення, тим більше

вихідна роздільна здатність і точніша модель, проте вона втрачає в швидкості. Multiplier – може приймати значення 1.01, 1.0, 0.75 або 0.50 (значення використовується лише архітектурою MobileNetV1). Це множник з плаваючою комою для глибини (кількість каналів) для всіх операцій згортки. Чим більше значення, тим більший розмір шарів і точніша модель за рахунок швидкості. QuantBytes – Цей аргумент керує байтами, які використовуються для квантування[12] ваги. Може приймати значення 4, 2 та 1. Вливає на якість розпізнавання та розмір моделі. Значення 4 – повнорозмірна модель, що має найвищу точність та розмір приблизно 90 мегабайт. Саме цими конфігураціями регулювалося відношення якості розпізнавання до швидкості під час експерименту.

Необхідно дослідити різницю в якості визначення поз при порівнянні та різницю в швидкості роботи мережі при різних налаштуваннях.

При порівнянні поз використовуються методи визначення відстані між векторами. Відрізняються лише методи побудови вектору та методи обчислення дистанції. Для дослідження були взяті 3 методи: косинусоїдної відстані, зваженої відстані та дистанції за обчисленими кутами. При реалізації перших двох методів вектори будуються зі значень координат для кожної ключової точки на тілі людини. Однак розміри зображень різні, відстань від камери та позиція у кадрі людини теж різна, тому необхідно нормалізувати ці значення. Перш за все, координати перераховуються таким чином, що точка початку координат знаходиться не з краю зображення, а з краю прямокутника, що оточує тіло людини. Наступним кроком йде L2 нормалізація вектору. Дистанція між отриманими векторами і буде характеризувати схожість поз на зображенні. В методі зважених дистанцій також враховується значення confidence, що вказує на точність отриманого передбачення положення ключової точки.

Для реалізації методу дистанції за обчисленими кутами нема потреби в нормалізації координат ключових точок, проте по іншому будується сам вектор. Вектор будується зі значень кутів між кінцівками, отриманих за формулою (1.7).

Таким кожну позу описує 10-вимірний вектор (положення голови в цьому методі не враховується).

Аналіз виконується за наступним алгоритмом: отримуємо описи поз за допомогою нейронної мережі в різних конфігураціях, отримуємо нормалізовані вектори, що порівнюються одне з одним за допомогою обох методів порівняння, результати зберігаються та використовуються для обчислення F1-показника. Цей показник обчислюється при різних порогових значеннях відстані між векторами: від 0.05 до 0.45 з кроком 0.05.

Таким чином ми отримуємо наступні результати:

Таблиця 3.1 – Результати дослідження методів порівняння поз

	ResNet50						MobileNetV1					
	Більша якість			Вища швидкість			Більша якість			Вища швидкість		
	CD	AD	WD	CD	AD	WD	CD	AD	WD	CD	AD	WD
$F_1^{0.05}$	48.1	50.3	60.2	47.3	46.4	43.9	0	12.2	18.1	12.4	11.3	12.9
$F_1^{0.10}$	55.6	54.8	56.3	54.2	52.1	56.1	0	23.7	27.6	21.9	19.8	22.0
$F_1^{0.15}$	55.4	56.2	58.7	59.8	60.0	60.8	32.9	34.3	35.6	26.7	27.1	30.8
$F_1^{0.20}$	58.7	57.3	65.4	61.0	60.5	62.6	30.7	35.8	36.8	28.7	33.2	37.4
$F_1^{0.25}$	60.1	57.8	64.9	62.2	62.4	64.1	33.0	34.1	33.4	30.0	33.1	37.1
$F_1^{0.30}$	58.2	56.2	59.0	60.4	60.8	59.9	33.2	32.0	31.2	31.3	32.5	35.6
$F_1^{0.35}$	55.0	55.3	52.3	51.7	53.2	51.5	30.5	30.6	28.9	31.4	29.0	34.8
$F_1^{0.40}$	49.3	50.1	47.5	44.9	49.4	45.4	29.9	30.1	26.9	29.6	28.1	32.3

Позначками CA, AD та WD позначаються результати для методів порівняння косинусоїдної дистанції, дистанції за обчисленими кутами та зваженої відстані, відповідно.

Виходячи з результатів аналізу отриманих даних можна зробити висновок, що найточніші результати порівняння поз дає метод зважених дистанцій з

пороговим значенням 0.25. Також результати наявно демонструють більш високу якість роботи мережі на архітектурі ResNet50.

Дані про швидкість роботи мережі представлено у таблиці 3.2. Обчислення виконані на процесорі 2.6 GHz Intel Core i7 та з 16GB оперативної пам’яті.

Таблиця 3.2 – Результати дослідження швидкості аналізу поз при різних налаштуваннях мережі

	ResNet50		MobileNetV1	
	High quality	High speed	High quality	High speed
Load time	55c	31c	5c	1c
Process time	255 мс	105 мс	222 мс	41 мс

Дослідження показали, що при порівнянні поз найбільшу вагу має якість визначення поз. Проте чим більша якість розпізнавання, тим нижча швидкість роботи, а для розпізнавання потокового відео необхідна висока швидкість.

Завдяки використанню сучасних методів визначення пози існує можливість реалізації такого проекту, для поз з низькою деталізацією, тобто для таких, де допускаються значні відмінності від оригіналу.

Оптимальним варіантом з точки зору швидкості та якості визначення в ході дослідження виявилася конфігурація моделі PoseNet, що базується на архітектурі ResNet50 оптимізована для швидшого виконання. Проте її швидкості недостатньо для аналізу швидких рухів об’єкту на відео, бо швидкість аналізу значно менша за необхідні 40мс.

Серед методів порівняння поз найкращий результат продемонстрував метод зважених дистанцій. Це пояснюється тим, що при порівнянні поз більша вага надається точкам, що були виявлені з більшою точністю і таким чином загальна поза порівнюється більш коректно, у той час як значення деталей нівелюється.

Основними недоліками такого підходу до порівняння поз є те, що результат напряму залежить від ракурсу зйомки. Таким однакові пози можуть виглядати

різними і навпаки при різних положеннях камери. Для того, щоб зменшити вплив ракурсу, необхідно будувати тривимірні моделі поз і порівнювати їх. Проте цей спосіб виконується значно довше і точність побудови тривимірних моделей поз за двовимірним зображенням значно нижча.

Тож виходячи з проведеного дослідження можна надати наступні рекомендації. При використанні бібліотеки PoseNet і необхідності порівняння результатів визначення поз, найточніший результат порівняння надає метод зважених дистанцій. Для визначення пози на потоковому відео найкращу швидкість показує конфігурація з використанням архітектури MobileNetV1, вона не може бути використана у випадках, коли є необхідність мати високу точність визначення. На потужних пристроях доцільно використовувати конфігурацію ResNet50, яка надає значно вищу точність визначення. Для реалізації додатку визначення правильності виконання фізичних вправ рекомендовано використовувати архітектуру ResNet50 з оптимізацією у бук швидкості виконання визначення пози. Але накладаються обмеження на швидкість виконання вправи, так як близько чверті кадрів не зможуть бути обробленими бібліотекою через низьку швидкість роботи.

4 АРХІТЕКТУРА ТА ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Проектування та моделювання програмного забезпечення відбувалося в три етапи. Перший – визначення потреб користувача, другий – моделювання архітектури програмного забезпечення, третій – моделювання окремих компонентів системи.

Взаємодію користувача з додатком описано за допомоги мови UML[15] у вигляді діаграми прецедентів (див. рис. 4.1).

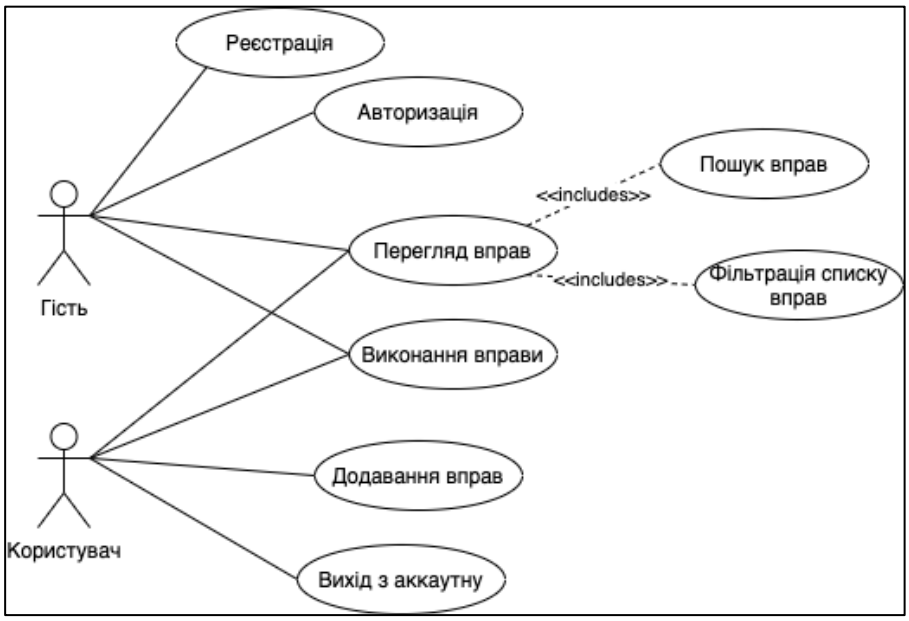

Рис. 4.1 – Діаграма прецедентів

На діаграмі представлено два типи користувачів: «Гість» та «Користувач». «Гість» – не авторизований у системі користувач, саме тому він має можливість авторизації або реєстрації і не має можливості виходу з системи. Функціонал додатку надає можливість пошуку необхідної вправи та її виконання незалежно від ролі. Проте додавати вправи може лише «Користувач».

Система складається з чотирьох ключових елементів: веб-додатку, серверного додатку, сховища даних та сховища файлів. Архітектура системи та

взаємодія компонентів наведені на діаграмі розгортання (див. рис. 4.2). Комунікація між сервером та клієнтом відбувається через HTTP-протокол. При розгортанні на віддалених серверах та запуску додатку для використання кінцевими користувачами необхідно використовувати протокол HTTPS, але для цього не потрібно вносити ніяких змін до початкового коду.

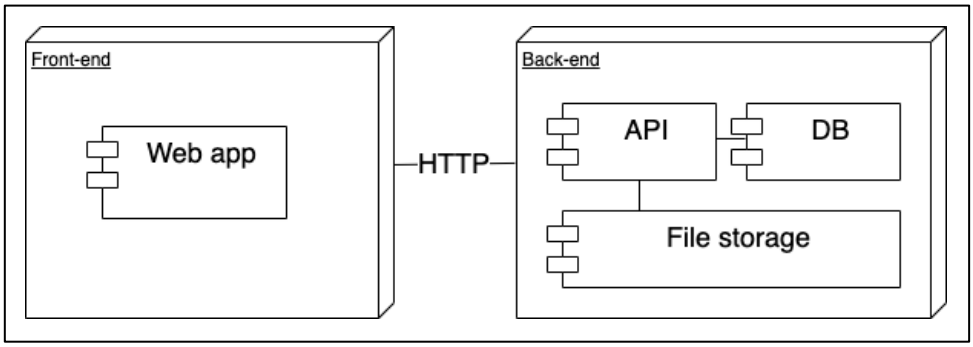

Рис. 4.2 – Діаграма розгортання

Комунікація веб-додатку з сервером проходить за допомоги серверного додатку, що надає API, побудований за архітектурним стилем REST. Серверний додаток, в свою чергу, зберігає дані у базі даних та файли у файловому сховищі. Особливу увагу варто приділити файлам. При завантаженні їх на сервер вони відправляються через API, він записує їх у файлове сховище та будує URL, за яким здійснюється доступ до цього файлу. Цей URL зберігається у базі даних при описі відомостей про файл. Для отримання файлу, веб додаток спочатку отримує інформацію від API, в отриманих даних знаходить URL та завантажує файли безпосередньо зі сховища. Таке розподілення зумовлено особливостями роботи з файлами та з даними, що їх описують.

Основним комплексним модулем веб-додатку є модуль перевірки виконання вправи. Його робота складається з двох етапів: ініціалізація модулю та перевірки виконання вправи. Ініціалізація починається з того, що бібліотека завантажує модель, відповідно до заданої конфігурації, що буде використовуватись методом визначення пози. Наступним кроком є визначення поз на еталонних зображеннях з використанням отриманого методу визначення поз та запуск циклу перевірки

виконання вправи. На рисунку 4.3 зображено діаграму послідовності, що відображує процес ініціалізації модулю.

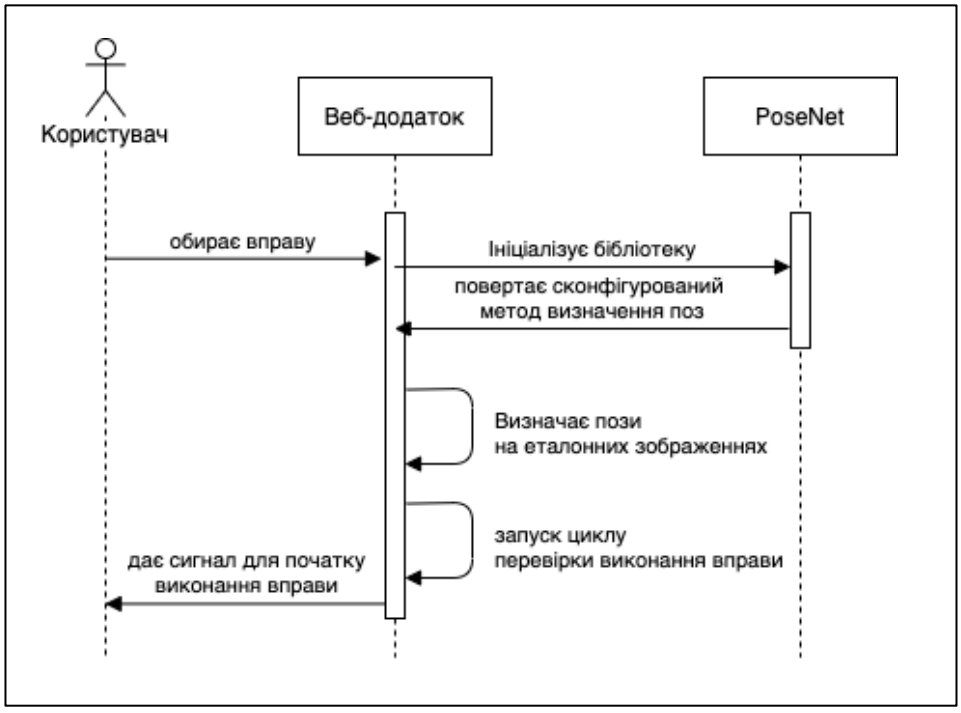

Рис. 4.3 – Діаграма послідовності

Цикл перевірки виконання вправи – це вже другий етап роботи модулю. Він полягає у тому, що кожен кадр з відео-камери користувача порівнюється з поточною позою з вправи. Кожна поза у вправі може мати три стани: «поточна», «пройдена» та «не пройдена». На початку виконання вправи всі пози мають стан «не пройдена». Поточною автоматична вважається перша зі списку не пройдених поз. Коли поточна поза співпадає з тою, що визначено з відео користувача, вона відмічається як «пройдена». Коли всі пози отримують статус «пройдена» вправа вважається виконаною. Цей процес відображено за допомоги діаграми/активності на рисунку 4.4.

При розробці додатку необхідно приймати до уваги, що швидкість визначення та порівняння пози з зображення на відео відбувається повільніше ніж змінюються кадри на відео. Якщо перевіряти кожен кадр, то за деякий час різниця

між кадрами, що оброблюються та кадрами, що надходять з камери користувача може бути занадто великою для коректної роботи.

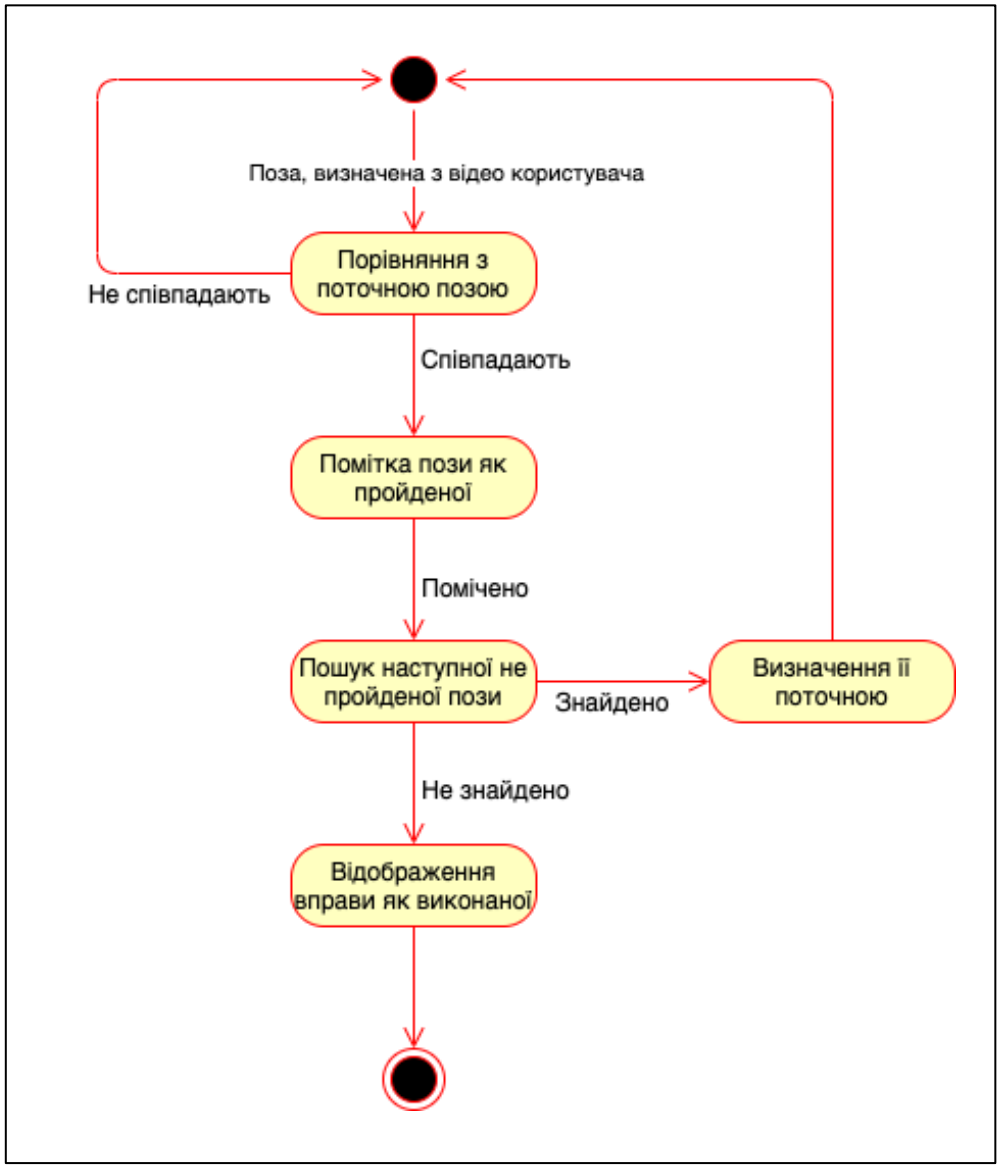

Рис. 4.4 – Діаграма активності

Тому процес визначення пози повинен виконуватись паралельно відображенню відео з камери та кожен новий цикл визначення та порівняння повинен починатися тільки після закінчення попереднього. Через це може виникати затримка при визначенні пози, але вона буде стабільною і рівномірною протягом усього часу роботи системи.

5 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ТА ТЕСТУВАННЯ

Задача серверної частини полягає у наданні доступу до даних. Всі обчислення виконуються веб-додатком, а сервер надає можливість виконання CRUD операцій для інформації про користувачів, вправи та для медіа-файлів.

Для збереження інформації використовується NoSQL база даних MongoDB [16]. Це документо-орієнтована база даних, що зберігає інформацію у вигляді JSON-об’єктів та не потребує притримуватись чіткої структури даних. Перевагою її використання є швидкість роботи read-операцій, які будуть виконуватись частіше ніж create-операції. Вибір цієї бази даних також зумовлено відсутністю великої кількості відношень між сутностями даних, які опрацьовує система.

Таких сутностей в системі 2: користувач та вправа. Користувач представлений в базі даних чотирма записами: id,nick, passwordHash, passwordSalt. Id генерується самою базою даних при доданні запису і є унікальним ідентифікатором, за яким можна знайти запис. Nick – обирається користувачем при створенні акаунту і також є унікальним для кожного користувача. PasswordHash та passwordSalt використовуються для збереження паролю у засекреченому вигляді у базі даних.

Інформація про вправи представлена у вигляді дочірніх колекцій опису користувача та складається з назви та колекції посилань до зображень. Посилання зберігається у тому порядку у якому повинні бути прийняті пози з зображень при виконанні вправи. Самі зображення зберігаються окремо у файловій системі.

Web API реалізовано мовою програмування TypeScript з використанням фреймворку NestJs. Фреймворк дозволяє оптимізувати розробку додатків, що надають REST API для своїх користувачів. Для реалізації авторизації використано бібліотеку PassportJs та стратегію JWT (JSON web token). Стратегія полягає у тому, що при авторизації на сервері за допомогу секретного ключа генерується токен, в якому є необхідна інформація про користувача (ідентефікатор та ім’я). Цей токен є

публічним і будь-хто, маючи його, може отримати цю інформацію. Він відправляється на клієнт і додається до кожного наступного запиту до сервера. Сервер завдяки секретному ключу валідує токен і знає, чи можна довіряти такому запиту чи ні. Час життя такого токену можна конфігурувати, але за замовченням він складає 15 хвилин, після чого процес авторизації необхідно проходити знов.

Для реалізації веб-додатку було використано бібліотеку ReactJs та мову програмування TypeScript. Завдяки цій бібліотеці інтерфейс будується з незалежних та блоків, що можуть бути використані повторно – компонентів. Основною перевагою є можливість використання розроблених компонентів при розробці мобільних додатків за допомогою фреймворку React Native.

Першою сторінкою, що бачить будь-який користувач є сторінка вибору вправи (див. рис. 5.1).

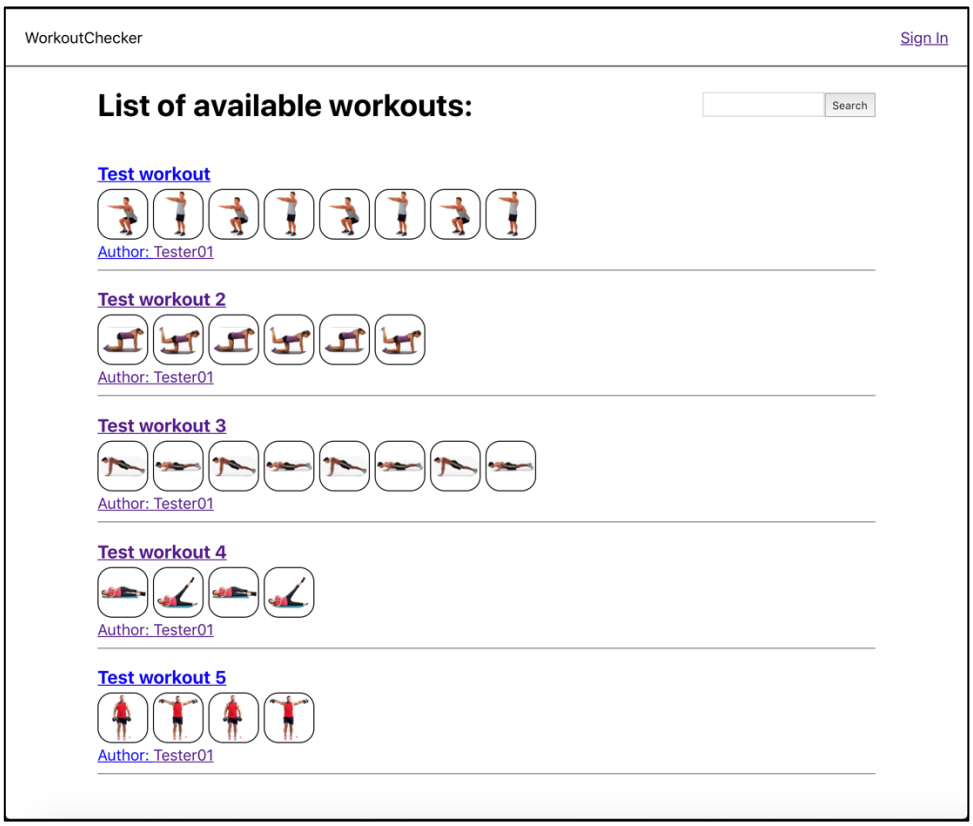

Рис 5.1 – Сторінка вибору вправи

Тут користувач може вибрати будь-яку вправу для виконання, шукати вправи за назвою або фільтрувати за користувачем, що створив вправу. Авторизований

користувач має змогу перейти на екран додавання вправи або вийти з акаунту, а не авторизований користувач має змогу перейти на сторінку авторизації або реєстрації. Різниця у навігаційних панелях полягає у тому, що гість бачить лише посилання на сторінку авторизації, а авторизований користувач також має посилання на сторінку створення вправ та сторінку зі списком його вправ.

Сторінка авторизації та реєстрації має 2 поля для вводу та дві кнопки. Поле для вводу унікального ім’я користувача та паролю. Кнопки відповідно для авторизації та реєстрації. При натисканні кнопки реєстрації виконується перевірка чи не зайняте вибране ім’я користувача. При натисканні кнопки авторизації виконується пошук користувача за наданим ім’ям та виконується перевірка відповідності наданого паролю. Інтерфейс цієї сторінки наведено на рисунку 5.3

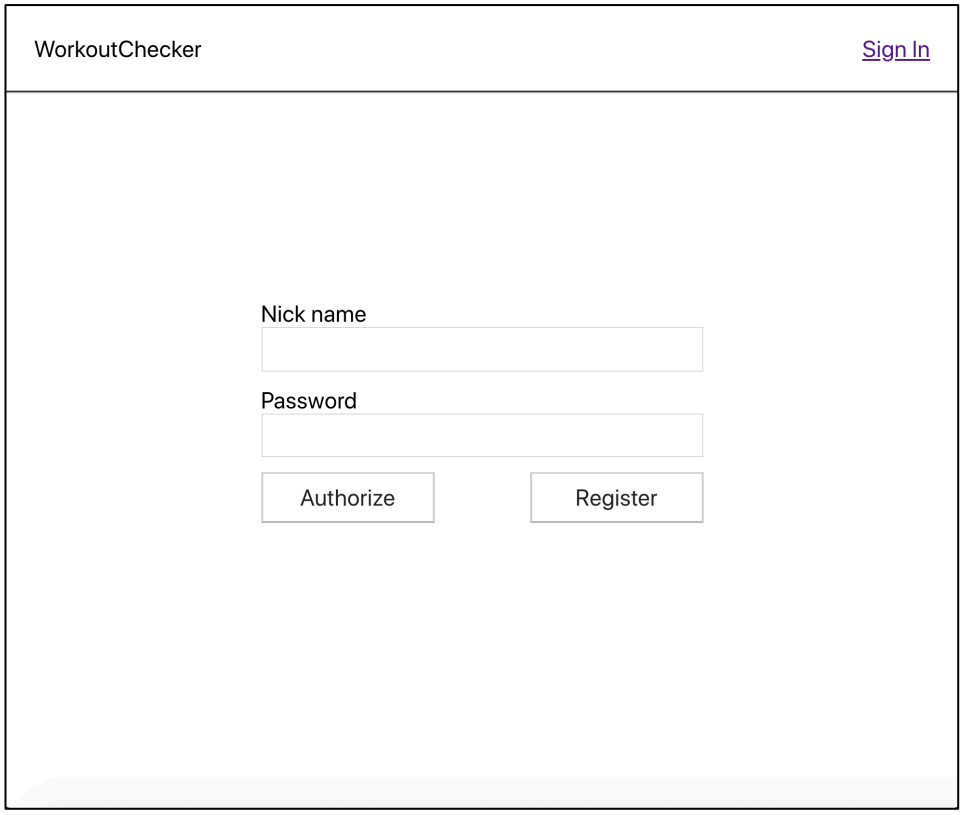

Рисунок 5.3 – Сторінка авторизації та реєстрації

Сторінка створення вправи (див. рис. 5.4) містить поле для вводу назви вправи, а також а також поля для вибору файлу з позою та порядкового номеру цієї пози у вправі. При натисканні кнопки «створити» вправа зберігається і стає

доступною іншим користувачам, а при натисканні кнопки назад відбувається перехід до попередньої сторінки.

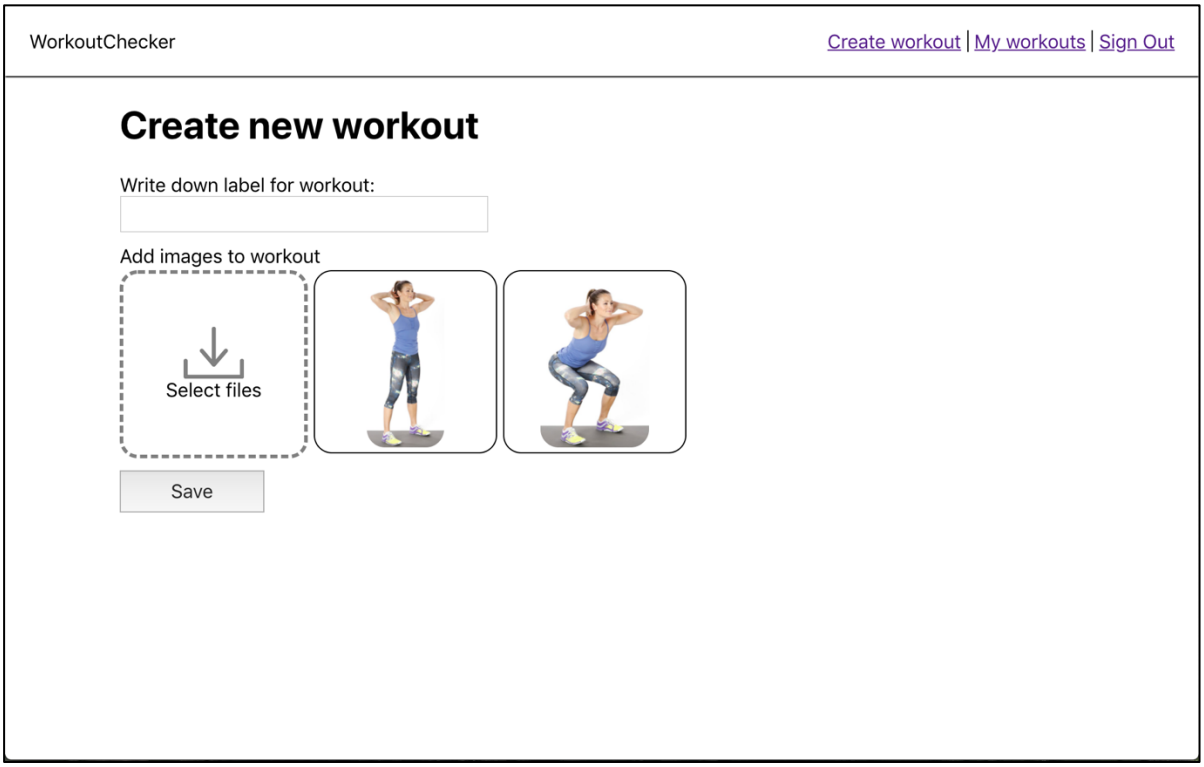

Рис. 5.4 – Сторінка створення вправи

Для розробки модулю визначення правильності виконання вправ також було використано мову програмування TypeScript. Середовищем виконання програмного коду є браузер. Джерелом потокового відео є камера, до якої користувачу необхідно надати доступ.

Безпосередня обробка та аналіз даних користувача виконується на його власному пристрої. Роботу модулю можна розбити на основні блоки: ініціалізація бібліотеки та отримання еталонних поз, отримання відео-потоку з камери користувача, визначення скелету людини на відео та порівняння пози людини з поточною позою.

На екрані користувач бачить зображення зі своєї камери та поруч з ним зображення зі списку поз у вправі (див. рис. 5.5). На зображені з камери відображено розмітку скелету людини, що був визначений програмою. Зеленим

відмічено пози, що вже було пройдено людиною, блакитним – ту, яку має прийняти користувач саме зараз та сірим – ті, що йдуть надалі.

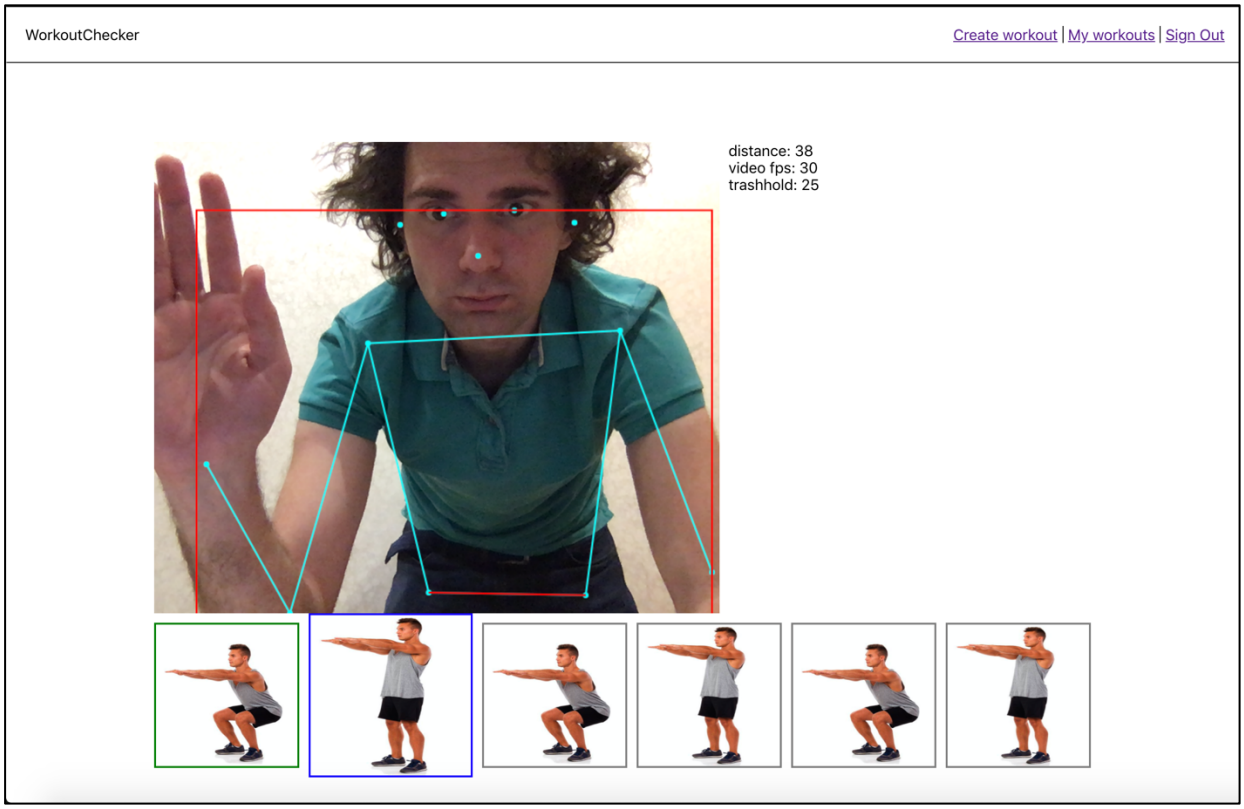

Рис 5.5 – Інтерфейс модулю перевірки виконання вправи

Для отримання та відображення відеопотоку з камери користувача використовується MediaStream API [17], що дозволяє працювати з окремими мультимедійними потоками даних, такими як аудіо та відео з різних джерел. Приклад коду, для отримання зображення з камери наведено далі:

```
async function setupCamera() {
  if (!navigator.mediaDevices) {
    throw new Error(
        'Browser API not available');
  }

  const video = document.getElementById('video');
  video.width = videoWidth;
  video.height = videoHeight;

  const mobile = isMobile();
  const stream = await navigator.mediaDevices.getUserMedia({
    'audio': false,
```

```
    'video': {
      facingMode: 'user',
      width: mobile ? undefined : videoWidth,
      height: mobile ? undefined : videoHeight,
    },
  });
  video.srcObject = stream;

  return new Promise((resolve) => {
    video.onloadedmetadata = () => {
      resolve(video);
    };
  });
}
```

Для отримання інформації про визначену позу людини необхідно перш за все ініціалізувати нейронну мережу необхідними параметрами. До цих параметрів належать architecture, outputStride, inputResolution, multiplier, quantBytes. За замовчуванням PoseNet завантажує архітектуру MobileNetV1 з множником 0,75. Виходячи з дослідження, описаного в попередніх розділах, оптимальними параметрами є архітектура ResNet50, outputStride – 16, quantBytes – 2.

Після загрузки мережа приймає на вхід джерело зображення, що може бути представлено у вигляді ImageData, HTMLImageElement, HTMLCanvasElement або HTMLVideoElement. Це робить бібліотеку дуже зручною у використанні і дозволяє розробнику не проводити додаткових маніпуляцій. Варто додати, що при наданні відео-елементів бібліотека обробляє лише один поточній кадр, а не неперервний потік. В результаті модель повертає колекцію об’єктів, що описують ключові точки на тілі людини, та загальну оцінку вірогідності, що роза була оцінена правильно. Приклад результату роботи аналізу зображення наведено на рисунку 5.6.

Код визначення пози виконується асинхронно. Слід зауважити, що середа виконання мови JavaScript однопоточна і тому обчислення не можуть бути виконані паралельно. Асинхронність необхідна для того, щоб довгі операції, що не потребують обчислень безпосередньо програмним кодом, такі як очікування відповіді від сервера, або читання інформації з файлу, не блокували виконання інших більш швидких задач та не призводили до заморожень інтерфейсу користувача. В якості механізмом паралелізації обчислень можна навести

використання воркерів – скриптів, що запускаються браузеров в окремому процесі та можуть взаємодіяти з сторінкою завдяки передачі повідомлень між процесами. Але цей механізм не було використано у бібліотеці.

Рис 5.6 – приклад результату визначення пози людини

Для побудови вектору необхідно спочатку привести значення координат до таких, що відповідають значенням системи координат, початком якої є обмежувальна границя тіла людини. Для цього необхідно знайти точку, що буде початком нової системи координат, що можна зробити завдяки функції getBoundingBox, яка приймає на вхід значення ключових точок та повертає значення maxX, maxY, minX, minY, що описують діагональні кути прямокутника, що обмежує людину на зображенні. Надалі координати ключових точок необхідно зменшити на значення відповідних мінімальних координат обмежувального прямокутника.

Нижче наведено функцію, що виконує обробку значень та L2 нормалізацію для координат ключових точок.

```
import { Keypoint, getBoundingBox } from "@tensorflow-models/posenet";

export function getNormalizedVector(keypoints: Keypoint[]): number[] {
    const {minX, minY} = getBoundingBox(keypoints);
    return keypoints.map(k => {
        return [
            ...L2normalization(k.position.x-minX,k.position.y-minY),
            k.score
        ]
    }).reduce((vector:number[],[x,y,score]:[number,number,number])=>{
        return [...vector, x, y, score];
    }, [])
}

function L2normalization(x: number, y: number): [number, number] {
    const length = Math.sqrt(x*x + y*y);
    return [x/length, y/length];
}
```

Таким чином отримуємо вектор готовий для порівняння, що є масивом чисел, які представляють послідовно значення координат x, y, а також значення точності виявлення контрольної точки.

При порівнянні необхідно мати два таких вектори, кожен з яких відображує значення для відповідного зображення. Наступний код виконує порівняння отриманих векторів.

```
function weightedDistanceMatching(poseVector1, poseVector2) {
  let vector1PoseXY = poseVector1.slice(0, 34);
  let vector1Confidences = poseVector1.slice(34, 51);
  let vector1ConfidenceSum = poseVector1.slice(51, 52);
  let vector2PoseXY = poseVector2.slice(0, 34);
  let summation1 = 1 / vector1ConfidenceSum;
  let summation2 = 0;
  for (let i = 0; i < vector1PoseXY.length; i++) {
    let tempConf = Math.floor(i / 2);
    let tempSum =
```

```
    vector1Confidences[tempConf]     *     Math.abs(vector1PoseXY[i]     -
vector2PoseXY[i]);
          summation2 = summation2 + tempSum;
        }

        return summation1 * summation2;
      }
```

Вектори poseVector1 та poseVector2 є векторами розміру 51, де елементи з 0 до 33 відображують значення координат 17-ти ключових точок, з 34 до 50 – значення точності знаходження відповідних точок, та значення номер 51 – суму значень точності для усього вектора. Чим меншим є результат виконання цієї функції, тим більш схожими є пози.

5.1 Тестування додатку та аналіз результатів

Аналіз пози виконується з меншою швидкістю, ніж змінюється зображення на відео. Це накладає обмеження на швидкість рухів при виконанні вправи, тому що деяка кількість кадрів буде пропущена при порівнянні. Для оцінки затримки між відео та визначенням пози розраховуються показники FPS – кількість кадрів на секунду, окремо для відео та для визначення поз. Середній показник для відео-зображення – 30 кадрів на секунду. Середній показник для визначення поз, отриманий на комп`ютері з процесором 2.6 GHz Intel Core i7 та 16 GB RAM – 13 кадрів на секунду. При тестуванні на мобільному девайсі iPhone 8 з частотою процесора 2.3 GHz та 2 GB RAM цей показник становить 8 кадрів на секунду. Цей показник дуже сильно відрізняється і не відповідає вимогам аналізу у реальному часі, бо для коректної роботи користувач повинен робити паузи у виконанні вправи, очікуючи порівняння поз.

При визначенні поз частіш за все помилки трапляються в ситуаціях, коли якісь з суглобів нечітко видно на зображенні. Приклад помилкового визначення пози наведено на рисунку 5.7.

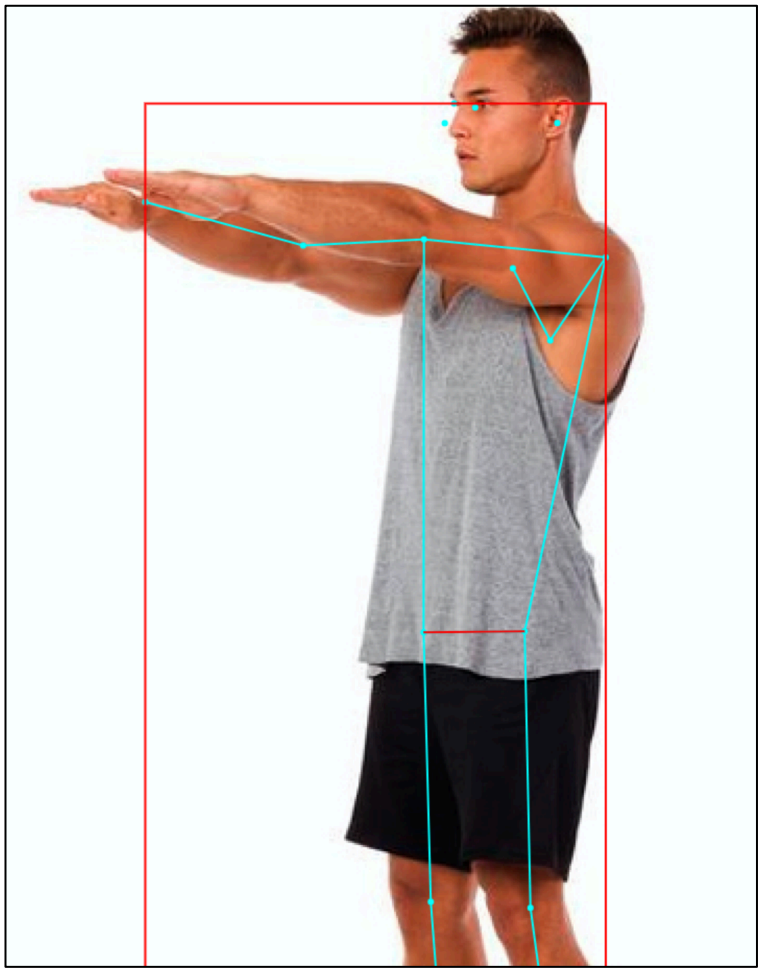

Рис. 5.7 – Приклад помилки при визначенні пози

На зображені можна побачити, що ті суглоби, які видно чітко та окремо, зокрема коліна, розпізнаються чітко, у той час як положення рук на зображені та на скелетоні не співпадають. При визначенні пози на відео це можна виправити, змістивши трохи руки чи якість освітлення, проте при визначенні пози, що отримується з зображення-еталону такі помилки критично впливають на якість роботи програми.

5.2 Рекомендації щодо покращення якості роботи додатку

При дослідженні методів порівняння було зроблено висновок, що використання архітектури ResNet50 з оптимізацією у бік швидкості надає оптимальні результати по відношенню якості порівняння поз до швидкості їх визначення. Цей висновок було зроблено з урахуванням якості порівняння і припущення, що її швидкість буде достатньої при виконанні повільних рухів. При занадто повільній швидкості визначення пози можна використовувати архітектуру MobileNetV1, що значно прискорює процес визначення, проте дуже негативно впливає на якість визначення. Порівняльну характеристику швидкості визначення поз архітектурами ResNet50 та MobileNetV1 на різних пристроях наведено у таблиці 5.1. Значення наведені у кількості кадрів на секунду.

Таблиця 5.1 – Результати дослідження швидкості визначення поз на потоковому відео на різних пристроях

	MacBook Pro (2.9HGz Intel Core i7)	iPhone 8 (2.3 HGz Apple A11 Bionic)	Sony Xperia X (1.8 HGz Qualcomm Snapdragon 650)
ResNet50	13	8	5
MobileNetV1	24	21	17

Для покращення точності роботи алгоритму порівняння поз необхідно покращувати якість визначення поз. В першу чергу необхідно покращувати якість визначення поз на еталонних зображеннях, бо якщо виникають помилки на цьому етапі, користувач не має можливості прийняти правильну позу і отримати бажаний результат.

Для покращення якості визначення поз на еталонних зображеннях необхідно покращити процес створення вправ. Якщо визначати пози на еталонних

зображеннях не в процесі роботи додатку, а заздалегідь, при створенні тренування, то це може підвищити точність порівняння, тому що в такому випадку швидкість визначення поз не має такої великої ролі і можна використовувати більш якісні конфігурації бібліотеки для більш точного визначення поз.

Наступним етапом у цьому напрямку стане реалізація редактора, який дозволяє редагувати автоматично визначенні пози на еталонних зображеннях і таким чином робити розмітку вручну.

Також для покращення якості роботи необхідно ретельно обирати зображення для тренування. Найкращі результати будуть у зображень, на яких не перетинаються кінцівки, на яких людина гарно освітлена та у яких монотонний фон, без надлишку дрібних елементів. Ті ж самі рекомендації стосуються зображення на відео.

ВИСНОВКИ

В результаті атестаційної роботи магістра було проведено аналіз існуючих бібліотек для визначення пози людини на зображенні, проведено дослідження можливостей бібліотеки PoseNet, створено датасет для аналізу якості порівняння поз, проведено дослідження і порівняння методів порівняння поз, а також розроблено систему перевірки правильності виконання вправ.

В результаті аналізу існуючих бібліотек для визначення пози людини на зображені з відкритим кодом було виявлено, що вони мають доволі близькі значення щодо якості визначення, проте бібліотека PoseNet має значні переваги з точки зору впровадження її до системи завдяки широкій підтримці мов програмування.

Серед методів порівняння поз найкращі результати продемонстрував метод зважених дистанцій завдяки тому, що він бере до уваги значення точності визначення ключової точки та при порівнянні надає більшу вагу тим точкам, що знайдено з більшою точністю.

При дослідженні можливостей бібліотеки PoseNet було визначено, що найкращі результати з точки зору швидкості та якості визначення поз дає конфігурація бібліотеки, що базується на архітектурі нейронної мережі ResNet50 з оптимізацією у бік швидкості. Але вона накладає обмеження на вправи з боку швидкості рухів. При швидких діях користувача система може не встигнути обробити зміни зображення, тому для визначення системою пози як правильної треба робити невеликі паузи у виконанні вправи. Також точність як методів визначення, так і методів порівняння надає можливість визначати пози лише з невеликою деталізацією. Наприклад, відмінність у положенні кистей будуть проігноровані системою.

Жодна з бібліотек та конфігурацій не надає точності вищої за 86%, тому для якісною роботи системи було підготовано ряд рекомендацій до вправ, зображень, та умов виконання. Чим менше прихованих суглобів на зображенні, тим точніше

визначення пози і тим точніша робота алгоритму порівняння, тому при створені вправи у системі необхідно обирати такий ракурс, що дозволяє зменшити перетин кінцівок та приховування ключових точок. Також для коректної роботи системи еталонні зображення та зображення на відео користувача повинні бути якомога чіткішими, гарно освітленими та не мати інших людей на фоні.

Аналіз роботи системи підтвердив результати досліджень і показав, що сучасні можливості дозволяють проводити достатньо точне визначення поз на потоковому відео лише для поз з низькою деталізацією та не великою швидкістю рухів на відео. Створення працездатної системи визначення правильності виконання вправ у формі веб-додатку можливе з усіма наведеними вище обмеженнями і використанням пристроїв, що мають високі технічні характеристики.

У подальшому необхідно дослідити можливість та ефективність реалізації подібної системи у вигляді нативних додатків для мобільних системи та ПК, а не в вигляді веб-додатку. Браузер накладає обмеження швидкості виконання, яких немає при написанні додатків. Також потрібно дослідити можливості крос-платформеного використання інших бібліотек. Але варто відзначити, що реалізація системи з відправкою відео на сервер і обробкою його віддалено не є доцільною через великі витрати часу на транспортному рівні.

Для покращення результатів створеної системи необхідно вдосконалити процес створення вправи з ціллю зменшення помилок при визначенні поз на еталонних зображеннях.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Shcherbakova, G.Y., Krylov, V.N., Bilous, N.V. Methods of automated classification based on wavelet-transform for automated medical diagnostics 2015 Information Technologies in Innovation Business Conference, ITIB 2015 – Proceedings 3, DOI: 10.1109/ITIB.2015.7355048

2. Poselet conditioned pictorial structures / L. Pishchulin [и др.] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. — 2013. — С. 588—595

3. Hubel, D. H.; Wiesel, T. N. (1968-03-01). "Receptive fields and functional architecture of monkey striate cortex". The Journal of Physiology. [Електронний ресурс] – Режим доступу: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1557912/

4. "Convolutional Neural Networks (LeNet) – DeepLearning 0.1 documentation" [Електронний ресурс] – Режим доступу: http://deeplearning.net/tutorial/lenet.html

5. Habibi, Aghdam, Hamed. Guide to convolutional neural networks: a practical application to traffic-sign detection and classification. Heravi, Elnaz Jahani, Cham, Switzerland.

6. Learning with Average Precision: Training Image Retrieval with a Listwise Loss [Електронний ресурс] – Режим доступу: https://arxiv.org/abs/1906.07589

7. Application of Convolutional Neural Network for Image Classification on Pascal VOC Challenge 2012 dataset [Електронний ресурс] – Режим доступу: https://arxiv.org/abs/1607.03785

8. Microsoft COCO: Common Objects in Context [Електронний ресурс] – Режим доступу: https://arxiv.org/abs/1405.0312

9. Білоус Н. В., Красов А. И., Власенко В. П. Видалення низькочастотної складової зображення з використанням медіанного фільтру // Журнал інженерних наук. 2016. Том 3 №2.

10. Jane Friedhoff, Irene Alvarado, Move Mirror: An AI Experiment with Pose Estimation in the Browser using TensorFlow.js [Електронний ресурс] – Режим

доступу: https://medium.com/tensorflow/move-mirror-an-ai-experiment-with-pose-estimation-in-the-browser-using-tensorflow-js-2f7b769f9b23

11. Pradnya Krishnanath Borkar , Marilyn Mathew Pulinthitha , Mrs. Ashwini Pansare, 2019, Match Pose – A System for Comparing Poses, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 08, Issue 10 (October 2019)

12. HigherHRNet: Scale-Aware Representation Learning for Bottom-Up Human Pose Estimation [Електронний ресурс] – Режим доступу: https://arxiv.org/abs/1908.10357

13. “OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields” [Електронний ресурс] – Режим доступу: https://arxiv.org/abs/1812.08008

14. Real-time Human Pose Estimation in the Browser with TensorFlow.js [Електронний ресурс] – Режим доступу: https://medium.com/tensorflow/real-time-human-pose-estimation-in-the-browser-with-tensorflow-js-7dd0bc881cd5

15. Крег Ларман, Застосування UML 2.0 і шаблонів проектування. 3-є вид. – М.: "Вільямс", 2006. – 736 с.

16. MongoDB database reference [Електронний ресурс] – Режим доступу: https://docs.mongodb.com/manual/reference/database-references/

17. Media Stream API [Електронний ресурс] – Режим доступу: https://developer.mozilla.org/en-US/docs/Web/API/MediaStream