ДОДАТОК А

Програмний код веб-застосунку з розпізнавання та контролю поз людини:

index.html

```
<!doctype html>
<html lang="en">
<head>
     <meta http-equiv="Content-Type" content="text/html; charset=utf-
8" />
     <meta name="viewport" content="width=device-width" />

     <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap
.min.css" integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
   <title>Fit Trainer</title>
</head>
  <body>
    <nav class="nav_bar">
       <div class="container">
       <a class="nav_bar link" href="#"><img src="https://icon-
       library.com/images/calories-icon/calories-icon.jpg" alt=""
       width="24" height="24">Train</a>
         </div>
     </nav>
     <h1 style="text-align: center;">Virtual fit train</h1>
     <div style="margin: 20px auto">
     <img src="{{ url_for('training') }}"></div>
     </body>
</html>
```

main.py

```
import cv2
import mediapipe as media_pipe
import numpy as num_py
import counting
import staging
```

```
from flask import Flask, render_template,
Response from training import Training

app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html')

def gen(training):
    while True:
        yield(b'--training.get_frame()\r\n'
              b'Content-Type: image/jpeg\r\n\r\n' +
              training.get_frame()+ b'\r\n\r\n')

@app.route('/training)
def training():
    return Response(gen(Training()))
app.run()
```

calculateAngle.py

```
import numpy as num_py
def angle_between_three_points(start, mid, end): =
num_py.array(start) num_py.array(mid)num_py.array(end)
radians    =    num_py.arctan2(end[1]-mid[1],    end[0]-mid[0])    -
num_py.arctan2(start[1]-mid[1], start[0]-mid[0])
    result = num_py.abs(radians*180.0/num_py.pi)

    if result > 180.0:
        result = 360 - result

    return result
```

counting.py

```
def init():
    global counting
    counting = 0
```

staging.py

```
def init():
    global staging
    staging = None
```

training.py

```
import cv2
import mediapipe as media_pipe
import numpy as num_py
from calculateAngle import angle_between_three_points
import counting
import staging
counting.init()
staging.init()

mp_draw =
media_pipe.solutions.drawing_utils
mp_pose = media_pipe.solutions.pose


class Training(object):
    def __init__(self):
        self.training = cv2.VideoCapture(0)

    def __del__(self):
        self.training.release()
    def get_frame(self):

        with mp_pose.Pose(min_detection_confidence=0.45,
min_tracking_confidence=0.45 , model_complexity=2) as pose:
            ret, frame = self.training.read()
            image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            image.flags.writeable = False
            results = pose.process(image)
            image.flags.writeable = True
```

```
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
    try:
    landmarks = results.pose_landmarks.landmark
       hip_right =
       [landmarks[mp_pose.PoseLandmark.RIGHT_HIP.value].x,
              landmarks[mp_pose.PoseLandmark.RIGHT_HIP.value].y
              ]
step_right =
[landmarks[mp_pose.PoseLandmark.RIGHT_SHOULDER.value].x,
landmarks[mp_pose.PoseLandmark.RIGHT_SHOULDER.value].y]
cubit_right =
[landmarks[mp_pose.PoseLandmark.RIGHT_ELBOW.value].x,
landmarks[mp_pose.PoseLandmark.RIGHT_ELBOW.value].y]
            hip_left                                          =
[landmarks[mp_pose.PoseLandmark.LEFT_HIP.value].x,landmarks[mp_p
ose.PoseLandmark.LEFT_HIP.value].y]
            step_left                                         =
[landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].x,
landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value].y]
            culbit_left                                       =
[landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].x,
landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value].y]
        left = calculateAngle(hip_left, step_left, culbit_left)
        right = calculateAngle(hip_right, step_right,
        culbit_right)
          cv2.putText(image, str(left),
                   tuple(num_py.multiply(
                   step_left, [640, 480]).astype(int)),
                   cv2.FONT_HERSHEY_COMPLEX, 0.7, (
                   255, 255, 255), 2, cv2.LINE_AA
                   )
        cv2.putText(image, str(right),
                   tuple(num_py.multiply(
                   step_right, [640, 480]).astype(int)),
                   cv2.FONT_HERSHEY_COMPLEX, 0.7, (
                   255, 255, 255), 2, cv2.LINE_AA
                   )
        if left < 30 and right < 30:
           staging.staging = "DOWN"
```

```
            if left > 90 and right > 90 and staging.staging ==
            'DOWN':
                staging.staging = "UP"
                count.count += 1
                print(count.count)

        except:
            pass
        cv2.rectangle(image, (425, 0), (640, 95), (255, 255, 255),
        -1)
        cv2.putText(image, 'Counting', (24, 10),
                    cv2.FONT_HERSHEY_COMPLEX, 1, (0, 0, 0), 1,
                    cv2.LINE_AA)
        cv2.putText(image, str(count.count),
                    (25, 75),
                    cv2.FONT_HERSHEY_COMPLEX, 2, (0, 0, 0), 2,
                    cv2.LINE_AA)
        cv2.putText(image, 'STATUS', (415, 14),
                    cv2.FONT_HERSHEY_COMPLEX, 1, (0, 0, 0), 1,
                    cv2.LINE_AA)
        cv2.putText(image, staging.staging,
                    (375, 50),
                    cv2.FONT_HERSHEY_COMPLEX, 2, (0, 0, 0), 2,
                    cv2.LINE_AA)
        mp_draw.draw_landmarks(image,
         results.pose_landmarks,
        mp_pose.POSE_CONNECTIONS,
        mp_draw.DrawingSpec(color=(0, 0, 255), thickness=5,
circle_radius=4),
        mp_draw.DrawingSpec(color=(255, 255, 0), thickness=5,
circle_radius=3)
)

ret, jpg = cv2.imencode('.jpg', image)
return jpg.tobytes()
```

ДОДАТОК Б

Графічна частина магістерської роботи

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ
ФАКУЛЬТЕТ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА УПРАВЛІННЯ
КАФЕДРА КІТС

Оцінки 3D пози і форми людини за допомогою штучних нейронних мереж

Магістрант гр. КІТм-21-1 Романов Р.Г.
Науковий керівник проф. Безсонов О.О.

Харків
2022

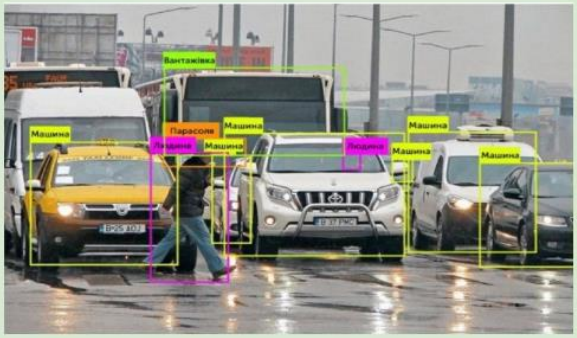
Мета роботи

Метою моєї роботи є створення веб-застосунку з використанням нейронної мережі для визначення кількісних та якісних параметрів активності людини. Штучні нейронні мережі є відгалуженням від глобального напрямку штучного інтелекту. Нейронні мережі можуть використовуватись для великого спектру задач, у тому числі для обробки зображень. Технологія обробки зображень для розпізнавання образів може забезпечити розпізнавання положення тіла людини з достатньо високою точністю, що дозволить відслідковувати її рухи та підраховувати кількість активності людини.

Задачі виконання дипломної роботи:

- провести аналіз предметної області;
- розгянути мову програмування та потрібних бібліотек для веб-застосунку;
- розробка програмного забезпечення для реалізації оцінки 3D поз та форм людини за допомогою штучних нейронних мереж.

Актуальність теми дослідження



Актуальність теми дослідження. Штучний інтелект – один із найголовніших напрямів розвитку сфери інформаційних технологій у сьогоднішній час. Штучні нейронні мережі походят від глобального напрямку штучного інтелекту. Нейронні мережі можуть використовуватись для різних задач, у тому числі для обробки зображень. Технологія обробки зображень для розпізнавання образів може забезпечити розпізнавання положення тіла людини, її пози з достатньо високою точністю, що дозволить підраховувати кількість активності людини та відслідковувати її рухи.

Стек технологій

- Python
- HTML + CSS + JS
- Flask
- MediaPipe

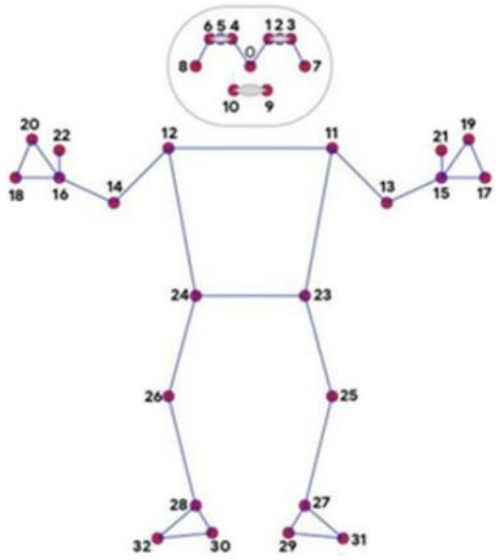
MediaPipe

MediaPipe – це проект Google, який надає «кросплатформне рішення з відкритим вихідним кодом, яке можна налаштувати для роботи з медіа в реальному часі та потоковим медіа». MediaPipe надає доступ до широкого спектру потужних моделей машинного навчання, створених з урахуванням апаратних обмежень мобільних пристроїв.
Разом із дуже простим у використанні API ці моделі спрощують процес розробки та скорочують час розробки проекту для багатьох програм, які покладаються на комп'ютерне бачення.

Ці моделі включають:

- ★ анатомічні моделі;
- ★ відстеження сітки обличчя;
- ★ відстеження пози;
- ★ відстеження рук.
- ★ об'єктні моделі;
- ★ виявлення/відстеження 2D об'єктів;
- ★ виявлення 3D об'єктів та оцінка пози.

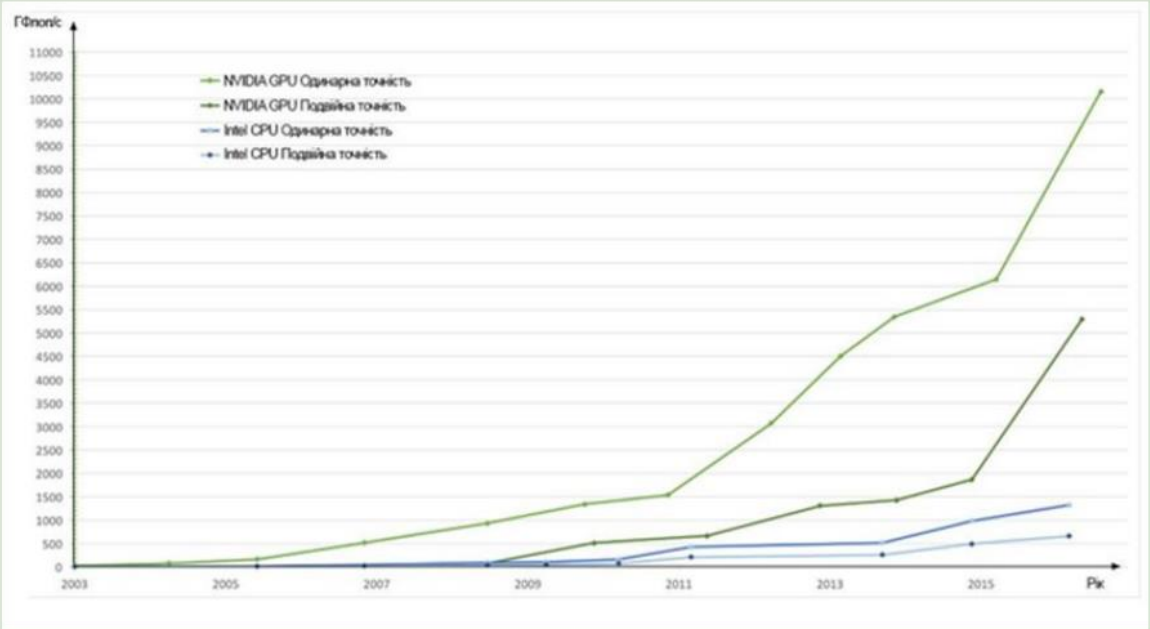
Список координат точок тіла для MediaPipe



Координати точок тіла

0) Ніс
1) Внутрішня ліва частина ока
2) Ліве око
3) Зовнішня частина лівого ока
4) Внутрішня права частина ока
5) Праве око
6) Зовнішня частина правого ока
7) Ліве вухо
8) Праве вухо
9) Ліва сторона рота
10) Права сторона рота
11) Ліве плече
12) Праве плече
13) Лівий лікоть
14) Правий лікоть
15) Ліве зап'ястя
16) Праве зап'ястя
17) Лівий мізинець
18) Правий мізинець
19) Вказівний палець лівої руки
20) Вказівний палець правої руки
21) Великий палець лівої руки
22) Великий палець правої руки
23) Ліве стегно
24) Праве стегно
25) Ліве коліно
26) Праве коліно
27) Ліва щиколотка
28) Права щиколотка
29) Ліва п'ята
30) Права п'ята
31) Індекс лівої стопи
32) Індекс правої стопи

Порівняння швидкості GPU та CPU



Навчання нейронної мережі полягає у пошуку оптимальних значень вагових коефіцієнтів нейронних зв'язків завдяки петлям зворотного зв'язку. Це також називається градієнтним зворотним розповсюдженням.

Схема процесу використання навчальної вибірки для навчання нейронної мережі:

ВИСНОВКИ

- В результаті виконання магістерської роботи був проведений аналіз застосування та використання штучних нейронних мереж на прикладі оцінки 3D поз та форм людини.
- За результатами проведених досліджень був створений веб-застосунок за допомогою мікрофреймворку Flask.
- Для оцінки 3D поз людини була використана бібліотека MediaPipe та у результаті було створено веб-застосунок для підрахунку кількості та якості виконання фізичних вправ.

Дякую за увагу :)