Pose-based Activity Recognition

Lena Mai, UNC Charlotte
Dr. Hongfei Xue, College of Computing and Informatics

UNIVERSITY OF NORTH CAROLINA CHARLOTTE

Introduction

Image/Camera-Based Human Activity

- Image/Camera-based human activity recognition has a variety of capabilities especially in healthcare, offering innovative solutions to improve patient care and safety.
- Application ranges from vital areas such as elderly fall detection, sport injury rehabilitation, and surgical procedures where real-time data and feedback is crucial.

Objectives

- **Extract Human Pose:** Retrieve human pose data from input images, analyzing both quantitative and qualitative aspects.
- **Develop a Pose-based Detection Program:** Create a real-time software application with OpenPose to detect and classify specific human postures accurately.
- **Analyze Pose Accuracy/Reliability:** Utilize JSON output data from OpenPose, focusing on scaled coordinate locations and confidence scores, to enhance pose detection accuracy.

Backgrounds

- **Human Pose Estimation:** This computer vision task involves identifying human body joints in images, crucial for applications such as interactive gaming and healthcare monitoring.
  - **Bottom-up Approach:** Detects key points before the person in the image. (e.g., OpenPose)
  - **Top-down approaches:** Detects the person in the image before the key points. (e.g., AlphaPose)

Tools

- **OpenPose:** Real-time detection framework used to identify human body positions from video and image data. Crucial for accurately detecting critical movements.
- **Python:** Favored for its extensive libraries supporting data analysis and machine learning. It scripts the integration of OpenPose outputs, analyzing JSON data to refine classification.
- **Docker:** Platform that maintains consistent developer environments by containerizing the application and its dependencies.

Methods

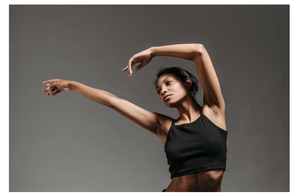
Steps Taken

1. **Setup:** Installed OpenPose demo on a Windows-based system equipped with an RTX 2060.
2. **Image Processing:** Ran personal data such as images, videos, webcam through OpenPose to evaluate its qualitative performance.
3. **Data Handling:** Analyzed BODY25 model outputs in OpenPose, evaluating through precision of joint coordinates between [0,1], where (0,0) represents the top-left corner and (1,1) the bottom-right of the image.
4. **Dataset Development:** Created a small dataset for three common poses and standardized their analysis using OpenPose.
5. **Script Development:** Based on analysis of JSON outputs, created a python script capable of identifying and reporting the detected poses.
6. **Environment Standardization:** Currently working on dockerizing OpenPose to ensure the application can be used across various development environments.

Results

Pose Accuracy with Python Script
Confidence Threshold: 0.5

Raised Arms (70% Accuracy)



- **Parameters:** Wrists and Elbows
- **Keypoint #:**
  - Right Wrist: 5
  - Left Wrist: 8
  - Right Elbow: 4
  - Left Elbow: 7
- **Analysis (y-coordinates):** Estimates if wrists are less value than the elbows.

Crossed Arms (40% Accuracy)

- **Parameters:** Wrists and Shoulders
- **Keypoint #:**
  - Right Wrist: 5
  - Left Wrist: 8
  - Right Shoulder: 3
  - Left Shoulder: 6
- **Analysis (x-coordinates):** Estimates if the right wrist is greater than right shoulder and left wrist is less than left shoulder.

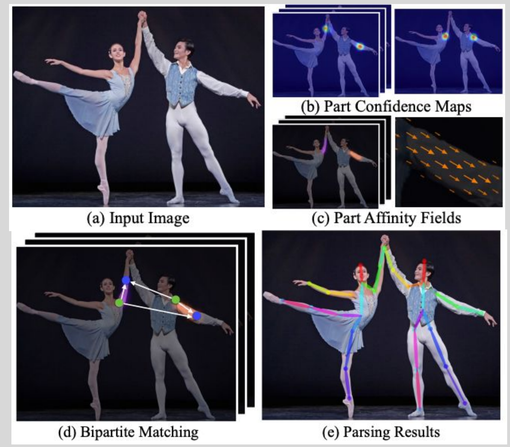
Arms on Hips (50% Accuracy)

- Parameters: Wrists and Hips
- Keypoint #:
  - Right Wrist: 5
  - Left Wrist: 8
  - Right Hip: 10
  - Left Hip: 13
- **Analysis (x-coordinate):** Needs proximity thresholds (0.1) that check if the wrist and hips' coordinates distance are below expected threshold.

JSON Output Example


```
{
  "version": 1.3,
  "people": [
    {
      "person_id": [-1],
      "pose_keypoints_2d": [
        0.332586, 0.189529, 0.852265,
        0.477512, 0.336561, 0.779027,
        0.26794, 0.347478, 0.636305
      ]
    }
  ]
}
```


Output displays the first three body points of a BODY25 model crossing their arms. JSON output is represented such by (x-coordinate, y-coordinate, confidence score)



Process of OpenPose Detection

Conclusions

- **Learning Foundation:** By developing a pose-based activity recognition program, it establishes basic understanding of real-time human pose detection.
- **Poses:** Poses vary based on body shapes and surroundings. Occlusions reduce the detection accuracy. While setting a threshold is important, we can potentially boost accuracy by exploring relationships among keypoints beyond just two.

Future Plans

- **Complex Poses:** Introduce more poses specifically related to harmful positions such as incorrect workout form and falling.
- **Database:** Analyze the poses using larger sample dataset such as COCO and to set a more accurate threshold.
- **Models:** Potentially explore other human pose estimation models such as AlphaPose.

References