Human body tracking and action recognition using neural networks

Project Report:

Devashish
SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
devdhaulakhandi@gmail.com

21BCE10076 Devashish Dhaulakhandi

ABSTRACT

The idea of computers being able to "look","see" and understand human actions has been central in the field of computer vision. However, the physical hardware,software and computing limitations have made it a challenge to do so effectively in a robust manner. The traditional approach of using Machine Learning has an alternative in Deep Learning models. The project aims to build such a deep learning based model. The model is divided into two phases, one, to capture time series data of the human action using a camera and use BlazePose, a lightweight convolutional neural network architecture for human pose estimation that is tailored for real-time inference. And second, to use the time series data to estimate the action performed by applying a sequential LSTM Model.

KEYWORDS

Neural Networks, CNN's, LSTM model, Blazepose , Computer Vision

1 INTRODUCTION

Human action recognition has many possible applications, surveillance, pose correction, detection of falls in case of elderly, detection of injuries during sports and many more, thus it covers many research topics in computer vision, detection of humans in the frame, pose estimation, tracking, and analysis and understanding of time series data.

It is also a challenging problem in the field of computer vision and machine learning.

For robust human action modelling, unlike feature representation in an image space, the representation of human actions in a video not only describes the appearance of the human(s) in the image space, but must also extract changes in appearance and pose.

The problem of feature representation is extended from two-dimensional space (only a photo) to three, including time-series data. From this perspective, most reviews of human action recognition are limited to approaches based on specific data, such as RGB data, depth data, or skeleton data.

Most of the already developed methods focus on human action recognition in RGB video data.

Supervised by NA.

In this project we tackle this task using a similar Vision-Based Activity Recognition: It uses a camera-based system to capture the behaviour of a subject.

The method consists of four steps: human detection, key-points extraction, key-points tracking and high-level activity evaluation. This image-based approaches uses a single camera to reconstruct the 3D human pose, to detect the coordinates of the joints and to extract the limbs key-points of the body in the frame. The image analysis is possible by isolating the human body from the background.This processed image sequence data is well segmented and contains only one action event.Further these coordinates, captured in a sequence (30 frames) are inputted to a Deep Learning LSTM model that makes the classification of the action performed.

2 BACKGROUND

Deep Learning:

Neural networks, or artificial neural networks, attempt to copy the way our brains work based on neurons through a system of data inputs, weights, and bias.

Deep neural networks consist of multiple layers of interconnected nodes or neurons,each building upon the previous layer to correct and optimize the prediction or categorization using the training data this progression of computations through the network is called forward propagation.

Another process called back-propagation uses certain cost based algorithms, like gradient descent, to calculate errors in predictions and then adjusts the weights and biases of the function by moving backwards through the layers in an effort to train the model.

Thus a neural network makes predictions and corrects for any errors accordingly. Over time, the method of prediction/classification becomes gradually more accurate.

Computer Vision:

It is a field of artificial intelligence (AI) that enables computers and systems to extract information from digital images, videos and other visual inputs.

If AI enables computers to think, computer vision enables them to see, observe and understand. However, implementations of computer vision need a lot of data.

Sub-domains of computer vision include scene reconstruction, object detection, event detection, video tracking, object recognition, 3D pose estimation, learning, indexing, motion estimation, visual surveying, 3D scene modelling, and image restoration.

Recurrent Neural Network(RNN):

Recurrent Neural Network(RNN) are a type of Neural Network where the outputs from previous step are fed as input to the current step. The main and most important feature of RNN is Hidden state, which remembers some information about a sequence. However they face the problem of Vanishing-Gradient and cannot function accurately as they do not retain information in the long term.

Blazepose:

Developed by Google AI, their approach provides human pose tracking by employing machine learning (ML) to infer 33, 2D landmarks of a body from a single frame.

BlazePose accurately localizes more keypoints in the frames. In addition, current state-of-the-art approaches rely primarily on powerful desktop environments for inference, whereas their method achieves real-time performance on mobile phones with CPU inference.

BlazePose achieves super-real-time performance, enabling it to run subsequent ML models, like face or hand tracking.

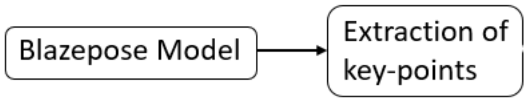


Figure 1: Blazepose

ML Pipeline for Pose Tracking :

For pose estimation, they utilize a proven two-step detector-tracker ML pipeline. Using a detector, this pipeline first locates the pose region-of-interest (ROI) within the frame. The tracker subsequently predicts all 33 pose key-points from this ROI, for video use cases, the detector is run only on the first frame. For subsequent frames they derive the ROI from the previous frame's pose key-points. This model only detects the location of a person within the frame and can not be used to identify individuals. The model is used in python through the library Mediapipe.

LSTM system:

It is a special kind of recurrent neural network capable of handling long-term dependencies.

A Long Short Term Memory Network is an advanced RNN, a sequential network, that allows information to persist. It is capable of handling the vanishing gradient problem faced by RNN.

At a high-level LSTM works very much like an RNN cell. The LSTM consists of three parts Just like a simple RNN, an LSTM also has a hidden state where H(t-1) represents the hidden state of the previous timestamp and H(t) is the hidden state of the current timestamp. In addition to that LSTM also have a cell state represented by C(t-1) and C(t) for previous and current timestamp respectively.

Thus an LSTM deals with sequential data and stores the information over long term as opposed to RNNs'.

3 METHODOLOGY

3.1 Design

The project consisted of two phases, generating the training data using videos of a subject performing some simple actions, and then

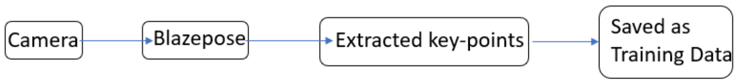


Figure 2: LSTM system training data

using that data to train a Deep Learning model and making the classification of the actions in real time.

3.2 Implementation

The key-points of the full human body are detected by the Mediapipe library and there are 33 of them. Each keypoint has an x,y,z (coordinated in the frame),and visibility attribute which are outputted as the results.

These are recorded and stored in a numpy array. In all for each action there are 33*4 = 132 key-points for each action. Thus this reduces the amount of data from millions of pixels to 30 (no. of videos) * 132 = 3960 data points for each action.

The training is done on a LSTM system of the structure:

Here is a flowchart depicting the process:

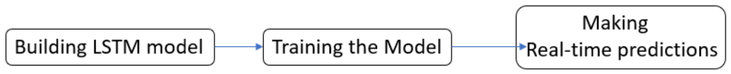
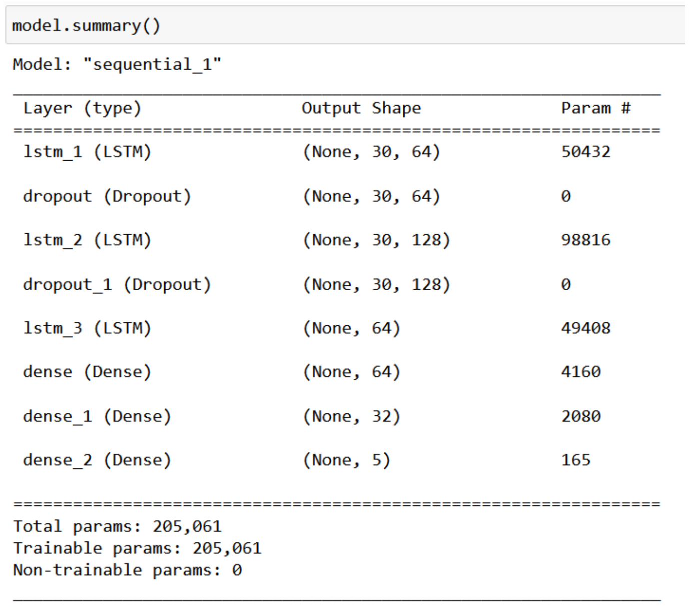


Figure 3: Implementation

The specifics of the libraries are given below:

- *Mediapipe*: The main CNN based key-points detection library for the human body that can detect and track them in real time in a robust manner.
- *Numpy*: The basic python library to work with arrays used for structuring the data.
- *Tensorflow,Keras,CUDA toolkit*: The python libraries used to access the gpu resources and use tensor cores to perform faster,more efficient calculations and train the models. The Deep learning model is directly built using these libraries.
- *cv2,Matplotlib*: They are python image/video processing and visualization libraries, cv2 is used to access the camera and image processing using filters etc while matplotlib is used to draw the animations/visual guides on the frames.
- *Optimization*: The LSTM model is optimised using Dropout layers to prevent overfitting.

Some Other Approaches: *The previous methods all dealt with higher dimensions as compared to our implementation. They use a variety data-sets like the COCO image data-set or the UCF 50 video data-sets. However since we required a simple 2D framework using only one camera for action classification, so we had to generate or own training data. The approach used can be said to be a skeleton based classification approach where we have not considered the background/clothes/light levels and other such subjective contextual details. A similar successful method is CNN based classification however it requires higher capacity computing resources that we did not have access to. It also deals with a large number of parameters and data*



```
model.summary()

Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_1 (LSTM)               (None, 30, 64)            50432

 dropout (Dropout)           (None, 30, 64)            0

 lstm_2 (LSTM)               (None, 30, 128)           98816

 dropout_1 (Dropout)         (None, 30, 128)           0

 lstm_3 (LSTM)               (None, 64)                49408

 dense (Dense)               (None, 64)                4160

 dense_1 (Dense)             (None, 32)                2080

 dense_2 (Dense)             (None, 5)                 165

=================================================================
Total params: 205,061
Trainable params: 205,061
Non-trainable params: 0
_________________________________________________________________
```

Figure 4: LSTM System Build

points. An image based classification model is also used however since it deals with still frames we avoided it as we needed a memory system that "remembers" the last 30 frames to make the classification(1-2 seconds).

4 CONCLUSION

Thus, we have build a human pose detector,tracker and actions classifier that can be used without expensive high end computing capacity hardware. The applications of this system are numerous and it solves a classic problem of ML/Deep Learning i.e. the computer understands what it "sees" a human doing. Such a system can be paired with the modern camera infrastructure in metro cities to make a centralized human action recognition system that has numerous applications:

- Fall detection using computer vision.
- Crowd surveillance and crime/riots detection.
- Accident detection in urban areas.
- Reducing injuries using estimation and detection in sports by correcting bad form.
- Body posture tracking for keeping good posture.
- etc and many more....:

5 REFERENCES/BIBLIOGRAPHY

- Tensorflow Based Image Classification using Advanced Convolutional Neural Network Pradumn Kumar, Upasana Dugal
- Image Classification with Classic and Deep Learning Techniques Òscar Lorente Corominas Ian Riera Smolinska Aditya Sangram Singh Rana
- ImageNet Classification with Deep Convolutional Neural Networks Alex Krizhevsky University of Toronto kriz@cs.utoronto.ca Ilya Sutskever University of Toronto ilya@cs.utoronto.ca Geoffrey E. Hinton University of Toronto hinton@cs.utoronto.ca
- Review of Image Classification Algorithms Based on Convolutional Neural Networks Leiyu Chen 1 , Shaobo Li 1,2,* , Qiang Bai 1 , Jing Yang 1,2 , Sanlong Jiang 1 and Yanming Miao 3
- Understanding the difficulty of training deep feedforward neural networks Xavier Glorot Yoshua Bengio DIRO, Universite de Montr ´ eal, Montr ´ eal, Qu ´ ebec, Canada
- Silhouette-based human action recognition using sequences of key poses Alexandros Andre Chaaraoui a, , Pau Climent-Pérez a , Francisco Flórez-Revuelta b
- Human Action Recognition and Prediction: A Survey Yu Kong · Yun Fu
- A Comprehensive Survey of Vision-Based Human Action Recognition Methods Hong-Bo Zhang 1,2,*, Yi-Xiang Zhang 1,2, Bineng Zhong 1,2, Qing Lei 1,2, Lijie Yang 1,2 ,Ji-Xiang Du 1,2,* and Duan-Sheng Chen 1,2
- A survey on vision-based human action recognition Uy Nguyn Văn
- BlazePose: On-device Real-time Body Pose tracking Valentin Bazarevsky Ivan Grishchenko Karthik Raveendran Tyler Zhu Fan Zhang Matthias Grundmann Google Research
- Learning Actionlet Ensemble for 3D Human Action Recognition Jiang Wang, Student Member, IEEE, Zicheng Liu, Senior Member, IEEE, Ying Wu, Senior Member, IEEE, and Junsong Yuan, Member, IEEE,
- Sequential Deep Learning for Human Action Recognition Moez Baccouche1,2, Franck Mamalet1, Christian Wolf2, Christophe Garcia2, and Atilla Baskurt2
- Latest Pose Estimation Realtime (24 FPS) using CPU | Computer Vision | OpenCV Python - YouTube (https://www.youtube.com/watch?
- Human Pose Estimation using opencv | python | OpenPose | stepwise implementation for beginners - YouTube (https://www.youtube.con
- Review: DeepPose — Cascade of CNN (Human Pose Estimation) | by Sik-Ho Tsang | Towards Data Science (https://towardsdatascience.cc deeppose-cascade-of-cnn-human-pose-estimation-cf3170103e36)
- Human Pose Estimation: Deep Learning Approach [2022 Guide] (https://www.v7labs.com/blog/human-pose-estimation-guide)
- TensorFlow Hub (https://tfhub.dev/google/movenet/multipose/lightning/1)
- CNN For Image Classification from scratch (https://www.analyticsvidhya.co classification-using-convolutional-neural-networks-a-step-by-step-guide/)
- Introduction to MediaPipe | LearnOpenCV (https://learnopencv.com/introdu to-mediapipe/)
- https://opencv.org/
- (https://www.tensorflow.org/?gclid=EAIaIQobChMI6oKd4qyi-gIVZplmAh3CJAgCEAAYASAAEgJj8$_{DBwE})https://www.tensorflow.or$
- https://keras.io/
- https://developer.nvidia.com/cuda-toolkit
- https://www.ibm.com/in-en/cloud/watson-studio?utm$_content = SRCWWp1 = Searchp4 = 43700052660685117p5 = egclid = EAIaIQobChMIy$–$7voq2i–gIV2gorCh3kgwGHEAAYAiAAEgJvg_DBwEgclsrc = aw.ds$