**LIST OF ABBREVIATIONS**

ML: Machine Learning

DL: Deep Learning

LSTM: Long-Short-Term-Memory-Network

RNN: Recurrent-Neural-Network

CNN: Convolutional-Neural-Network

AI: Artificial-Intelligence

ROI: Region-Of-Interest

Params: Parameters

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **TITLE** |  |

Figure 1: The picture presents a chart, organized in a hierarchical tree of currently used methods.

Figure 2: Blazepose in real time

Figure 3: Blazepose in real time

Figure 4: Key-points Extraction from Blazepose

Figure 5: Blazepose Architecture

Figure 6: LSTM Architecture

Figure 7: Flowchart for generation of training data

Figure 8: Libraries/Packages used

Figure 9: LSTM architecture

Figure 10: Deep Learning model architecture summary

Figure 11: Flowchart for Proposed Method

Figure 12: Stage one module formulation

Figure 13: Full System design and architecture

Figure 14: Real time Classification

Figure 15: Real time Classification

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO.** | **TITLE** |  |

Table 1: Categorical\_accuracy and loss of model based on training data

Table 2: Classification Accuracy\_score on Training data

Table 3: Applications

**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 project is divided into three stages, 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. The key-points obtained from the model are recorded and saved to use as training data for certain actions performed by the subject. Second, the LSTM is built and trained on the generated data, the weights are saved and this brings us to stage three, to use the time series data to estimate the action performed by applying the trained sequential LSTM Model.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | List of Abbreviations  List of Figures  List of Tables  Abstract | iv  v  vi  vii |
| 1 | **INTRODUCTION** 1.1 Introduction 1.2 Motivation for the work  1.3 Problem Statement  1.4 Techniques  1.6 Summary |  |
| 2 | **LITERATURE SURVEY**  2.1 Introduction  2.2 Core area of the project  2.3 Existing Algorithms  2.3 Observations from literature Survey |  |
| 3 | **SYSTEM ANALYSIS**  3.1 Introduction  3.2 Disadvantages/Limitations in the existing system  3.3 Proposed System |  |
| 4 | **SYSTEM DESIGN AND IMPLEMENTATION**  4.1 Introduction  4.2 Stage 1 design & module implementation  4.3 Stage 2 design & module implementation  4.4 Summary |  |
| 5 | **PERFORMANCE ANALYSIS**  5.1 Performance Analysis  5.3 Summary |  |
| 6 | **CONCLUSION AND FUTURE WORK**  6.1Introduction  6.2 Constraints of the System  6.3 Future Enhancements  6.4 Conclusion |  |
| 7 | References , Bibliography |  |

**INTRODUCTION**

* 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.

**1.2 Motivation for the work**

Human action recognition 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. This problem is the topic of ongoing research and development for deep learning and AI/ML enthusiasts around the world. The applications of a software that can “see” and understand the actions being performed by any human are endless. Thus, this motivated us to build/ propose our own small-scale system to do the same with available resources and hardware.

**1.3 Problem Statement**

The problem of feature representation is extended from two-dimensional space (only a photo) to include time-series data(a video). In this project we tackle this task using a similar Vision-Based Activity Recognition: It uses a camera-based system to capture the behavior of a subject. Our scope is defined up to this stage, to train a DL model to capture and accurately classify the actions of a human subject being performed in front of the camera.

**1.4 Techniques**

Vision-based action recognition and prediction from videos are tasks, where action recognition is to infer human actions (present state) based upon complete action executions. This task has become a particularly prevalent topic recently because of its’ explosively emerging real-world applications, such as visual surveillance, autonomous driving vehicle, entertainment, and video retrieval, etc.

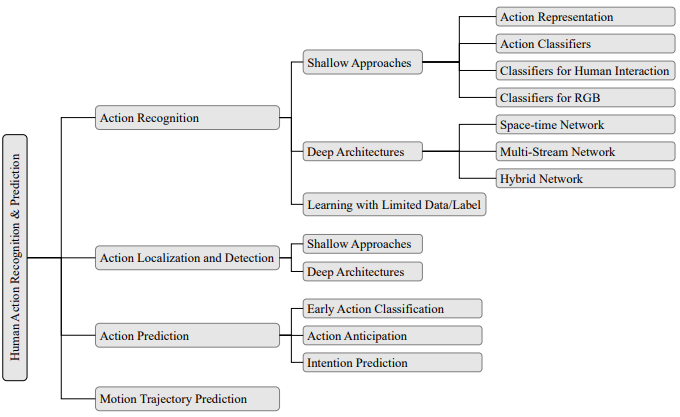


Fig 1: The picture presents a chart, organized in a hierarchical tree of currently used methods.

Action Representation:

(Yu Kong · Yun Fu 22) The first and the foremost important problem in action recognition is how to represent an action in a video. Human actions appearing in videos differ in their motion speed, camera view, appearance and pose variations, etc., making action representation a really challenging problem. A successful action representation method should be efficient to compute, effective to characterize actions, and can maximize the discrepancy between actions, in order to minimize the classification error.

Holistic Representations:

(Hong-Bo Zhang et al 2019) Human action in a video generates a space-time shape in the 3D volume. This spacetime shape encodes both spatial information of the human pose at various times, and dynamic information of the human body. Holistic representation methods capture the motion information of the entire human subject, providing rich and expressive motion information for action recognition. However, holistic representations tend to be sensitive to noise. It captures the information in a certain rectangle region, and thus may introduce irrelevant information and noise from the human subject and cluttered background.

Local Representations:

(Simonyan et al 2014) Local representations only identify local regions having salient motion information. Successful methods such as space-time interest points and motion trajectory have shown their robustness to translation, appearance variation, etc.

Action recognition from RGB-D videos has been receiving a lot of attention due to the advent of the cost-effective Kinect sensor. RGB-D videos provide an additional depth channel compared with conventional RGB videos, allowing us to capture 3D structural information that is very useful in reducing background noise and simplifying intra-class motion variations.

Action recognition using Deep Architectures:

(Tran, D et al 2015 )Although great success has been made by global and local features, these hand-crafted features require heavy human labor and domain expert knowledge to develop effective feature extraction methods. In addition, they normally do not generalize very well on large datasets. In recent years, feature learning using deep learning techniques has been receiving increasing attention due to their capability of learning powerful features that can be generalized very well. The success of deep networks in action recognition can also be attributed to scaling up the networks to tens of millions of parameters and massive labeled datasets.

**1.6 Summary**

One of the ultimate goals of artificial intelligence research is to build a machine that can accurately understand humans’ actions and intentions, so that it can better serve us. Important applications including visual surveillance, entertainment, and video retrieval also need to analyze human actions in videos. In the center of these applications is the computational algorithms that can understand human actions. (Güler et al 2017 )Like the human vision system, the algorithms ought to produce a label after observing the entire or part of a human action execution. Building such algorithms is typically addressed in computer vision research, which studies how to make computers gain high-level understanding from digital images and videos.

**LITERATURE REVIEW**

**2.1 Introduction**

Although widely used in many applications, accurate and efficient human action recognition remains a challenging area of research in the field of computer vision. Most recent methods have focused on narrow problems such as human action recognition methods using depth data, 3D-skeleton data, still image data, spatiotemporal interest point-based methods, and human walking motion recognition.

**2.2 Core area of the project**

(Zhang et al 2022 )The key to good human action recognition is robust human action modeling and feature representation. Feature representation and selection is a classic problem in computer vision and machine learning. Unlike feature representation in an image space, the feature representation of human action in video not only describes the appearance of the human(s) in the image space, but must also extract changes in appearance and pose.

**2.3 Existing Systems**

From the perspective of data type, research on human action recognition can be divided into methods based on color (RGB) data and methods combining color and depth data (RGBD).The human action recognition approaches for these data, following the progress of machine learning research, can be categorized as either hand-designed features with machine learning methods or end-to-end deep learning algorithms. Regardless of data type and computing method, the core aim is to extract robust human action features. Many action features have been proposed for RGB data, such as spatiotemporal volume-based features , spatiotemporal interesting point features , and joint trajectory features . (Zhang et al 2022 ) However, factors such as camera movement, occlusion, complex scenes, and the limitations of human detection and pose estimation methods limit the performance of human action representation and recognition based on handcrafted features. Because depth data are stable with respect to changes in environment and background and allow objects to be quickly segmented according to depth, the application of depth sensors enables real-time, robust human pose estimation. Human action recognition methods based on depth information and skeleton sequences demonstrate high recognition accuracy and low time complexity. These methods are popular in human action recognition research. However, the accuracy and cost of depth sensors mean that depth- and skeleton-based action recognition methods are currently only applicable over limited ranges and in specific environments. There are three types of commonly used depth cameras: triangulation (with two camera views), time-of-flight (TOF) cameras, and structured-light-based cameras. Structured-light and TOF-based depth sensors are easily affected by light, with large errors and low precision in outdoor environments. The cost of the two-camera system is lower, but the depth information calculation has higher complexity and cannot be applied in darker environments. In addition, there are other sensors that can be used to measure depth, such as laser scanners; however, these devices are expensive and unsuitable for video surveillance and home monitoring. Unlike handcrafted action features, deep learning methods perform well with regard to automatic feature learning from images. This provides a new insight into human action recognition, and many researchers have attempted to use deep learning methods to extract action features from RGB, depth, and skeleton data. Such data are applicable to multimodal feature learning from deep networks, such as appearance/image information, optical flow sequences, depth sequences, and skeleton sequences. Deep learning networks can learn human action features from single-mode data or multimodal fusion data. As the appearance sequence and optical flow sequence are relatively easy to obtain, most deep learning methods adopt the appearance sequence and optical flow sequence as their input, with few depth- and skeleton-based techniques.

However, recent high-efficiency multi-person pose estimation methods based on deep learning have drawn increased attention to human action feature learning based on skeleton sequences, and this is now a prominent research topic in the field of human action recognition .

**2.3 Observations from the literature Survey**

A summary of these methods reveals that the design of the interaction feature mainly follows a set of principles:

(1) Local interaction features should be dense enough to represent information at various locations in the image.

(2) The model of interaction between the human and the object(s) is based on the structure of body parts.

(3) The core of the interaction model is the co-occurrence and position relationship between the human body and the object(s).

(4) Features with higher discriminative power are selected from dense features.

**SYSTEM ANALYSIS**

**3.1 Introduction**

The systems used in Human-Action-Recognition are software and hardware based.

The traditional approaches consist of a hardware-based sensor data to be used as training data, a software-based ML/AI/DL models to be used for the final real time or test data classification.

The existing systems either don’t use cameras as a sensor, in which case they go out of scope of our project. The sensors these systems used are accelerometer, gyroscope etc. to get the real time speed, time series data and behavioral data of the subject. E.g.: Googles’ fitness tracker, Smart watches etc.

We are more concerned with the other classification systems that use camera, capture the temporal changes in the pose of a subject and then make a classification.

**3.2 Limitations in the existing system**

1, The current systems rely on pre-built/already compiled datasets, however we could not do the same as we needed real time tracking and classification.

2, The traditional neural approaches use CNN (Convolutional Neural Networks) and other similar feature modeling and capturing methods.

3, These approaches however, require many computational resources and are not very robust. In case of lack of the required infrastructure the models may not deliver results on time.

4, In case the traditional models are trained on data from multiple cameras, then the data would wary from shakiness, camera positions, distance from subject, specifications, fps etc.…

Some Other traditional approaches: The previous methods all dealt with higher dimensions as compared to our implementation. They used a variety of data sets like Hollywood2 ,HMDB51, Olympic Sports, UCF50, UCF101, Kinetics, MSR-Action3D, MSR-Daily Activity, Northwestern-UCLA, UTD-MHAD, RGBD-HuDaAct , NTU RGB+D etc. 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 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).

Due to these factors and the fact that we required a simple to implement system that could be deployed on any camera and system of varied specifications, we developed our own method.

**3.3 Proposed System**

The proposed method leverages the efficiency and novelty of neural networks in classification tasks and approaches the problem in a direct way. The system used is a RGB data input but skeleton based neural classification model that uses and LSTM (Long Short-Term Memory System) to carry out real time execution and classification of the actions of a human subject. The system is built in the Python language and the packages/concepts used and their specifications are mentioned below.

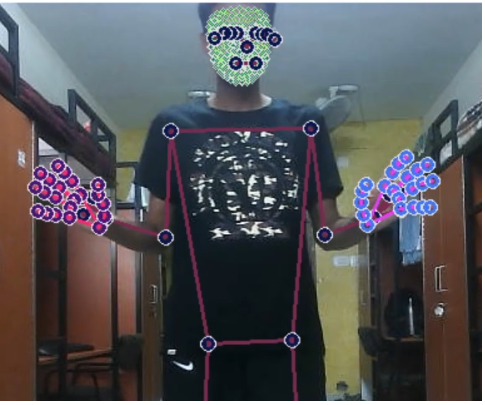


Fig 2 Blazepose in real time Fig 3 Blazepose in real time

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 key-points in the frames.

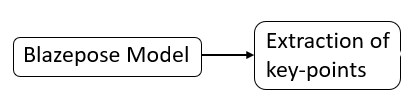


Fig 4 Key-points Extraction from Blazepose

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, 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 cannot be used to identify individuals. The model is used in python through the library Mediapipe.

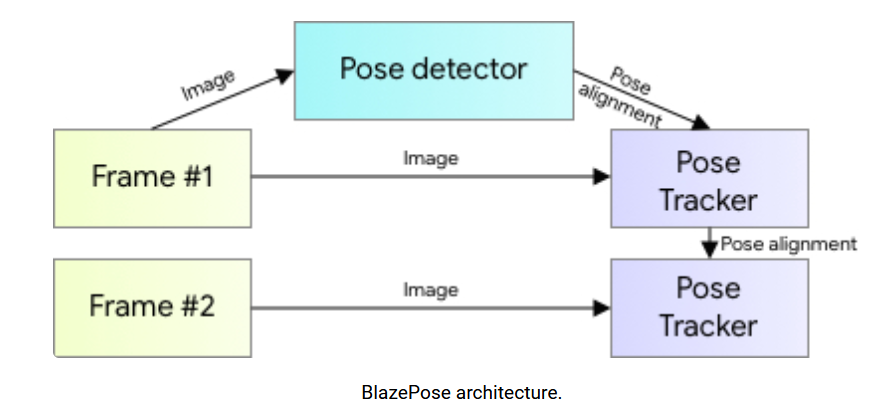


Fig 5

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.

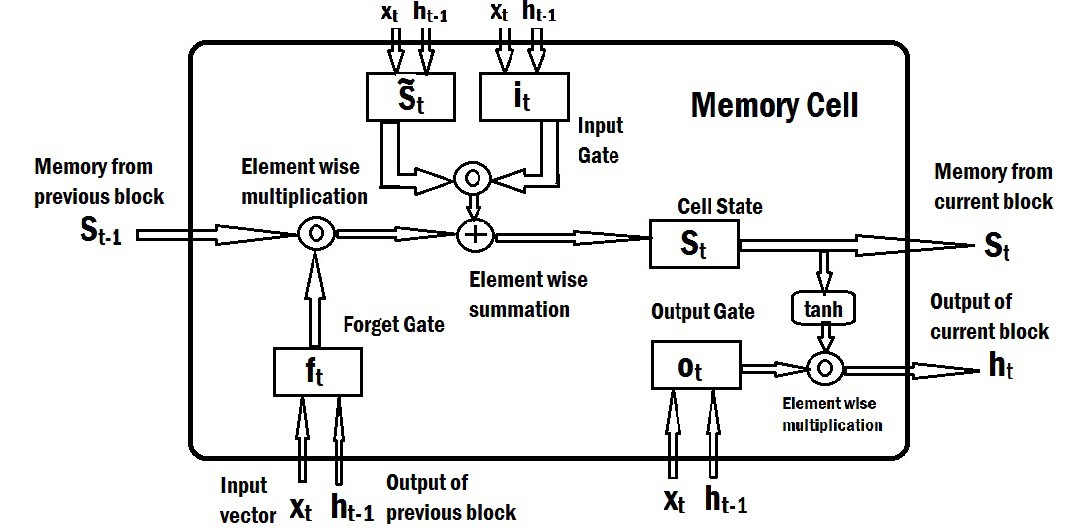
The LSTM consists of three parts just like a simple RNN, an LSTM also has a hidden state where H(t-1) represents the hidden state of the previous timestamp and H(t) is the hidden state of the current timestamp. In addition to that LSTM also have a cell state represented by C(t-1) and C(t) for previous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

Figure 6 LSTM Architecture

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

**SYSTEM DESIGN AND IMPLEMENTATION**

**4.1 Introduction**

The project consisted of two phases, generating the training data using videos of a subject performing some simple actions, and then using that data to train a Deep Learning model and making the classification of the actions in real time.

Implementation:

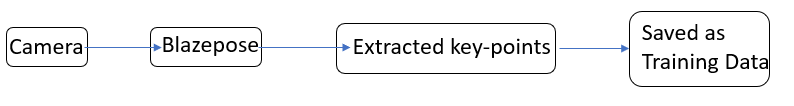
The key-points of the full human body are detected by the Mediapipe library and there are 33 of them. Each key point has an x,y,z (coordinated in the frame),and visibility attribute which are outputted as the results. These are recorded and stored in a numpy array. In all for each action there are 33\*4 = 132 key-points for each action. 

Fig 7 Flowchart for generation of training data

Thus, this reduces the amount of data from millions of pixel values to 30 (no. of videos) \* 132 = 3960 data points for each action.

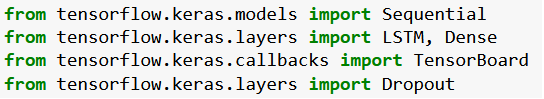
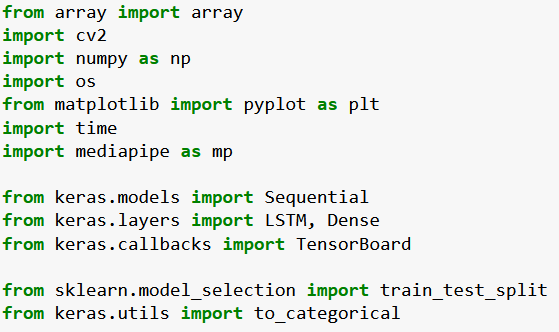


Fig 8 Libraries/Packages used

The specifics of the libraries are given below:

• Mediapipe: The main ensemble-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.

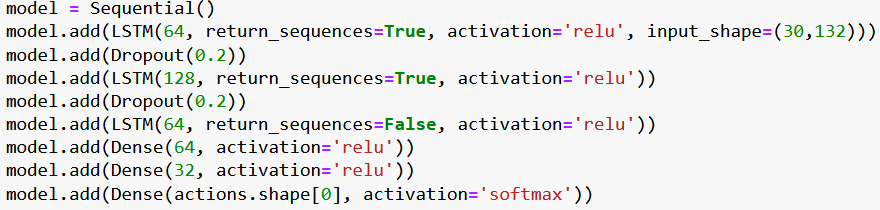


Fig 9 LSTM architecture

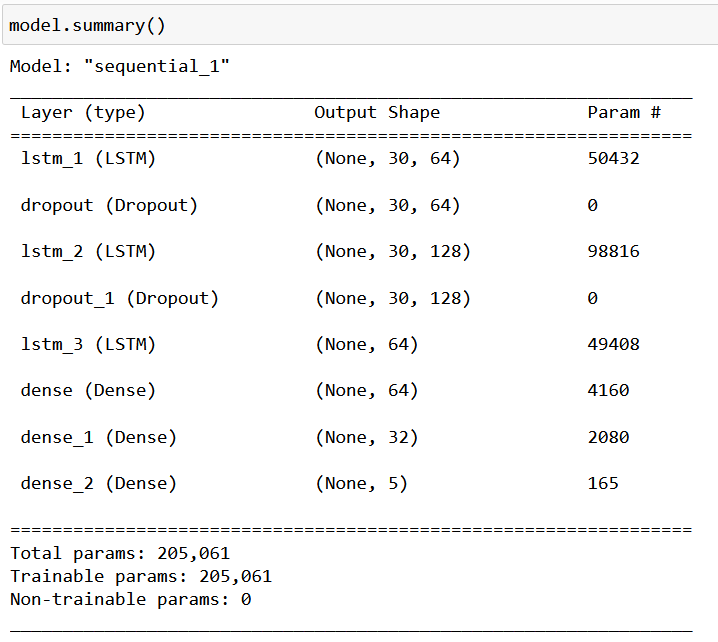


Fig 10 Deep Learning model architecture summary

• 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 optimized using Dropout layers to prevent overfitting.

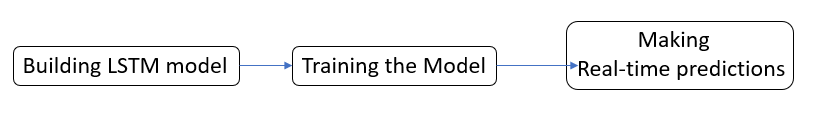
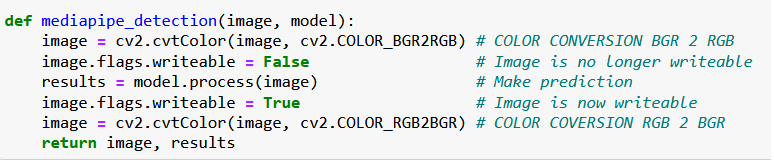


Fig 11 Flowchart for Proposed Method

**4.2 Stage 1, Module design & implementation**

This stage is concerned with the generation of the training data for the DL model to “learn” about the actions that it is going to be used to classify.

The opencv library is used in conjunction with Mediapipe to get a real time key-points detection and tracking module. For each action class 30 videos of 30 frames each are recorded and the key-point sequences are stored in numpy arrays. This is the training data.



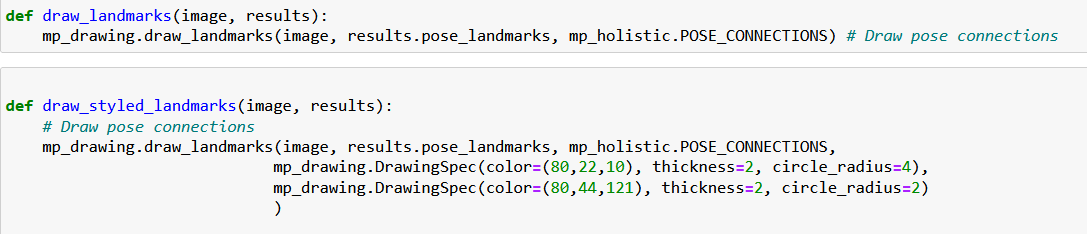


Fig 12 Stage one module formulation

**4.3 Stage 2, Module design & Implementation**

In this stage we build our LSTM model and use the generated data to train it on the recognition of classes. Thus, the training goes on for 100’s of epochs until the desired accuracy of about 95% with loss value of 0.15 is obtained. Then the DL model weights are saved and finally the trained model is used to make the classifications in real time.

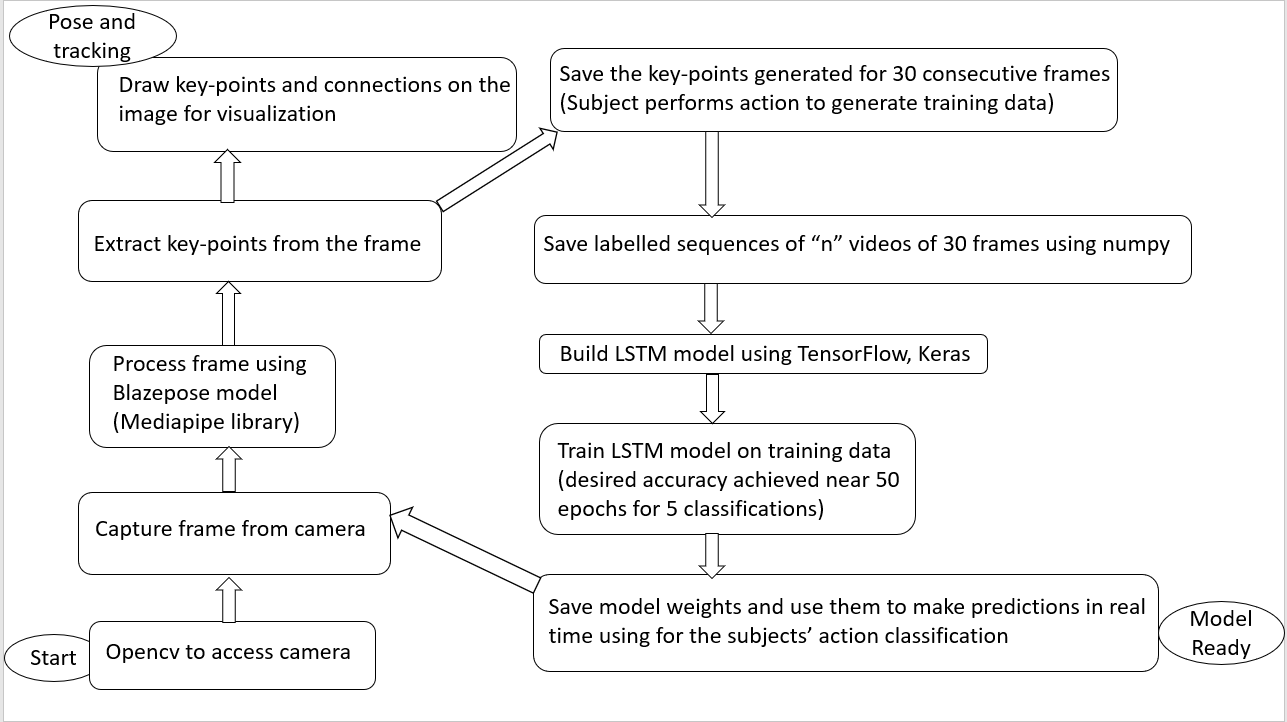


Fig 13 Full System design and architecture

**4.4 Summary**

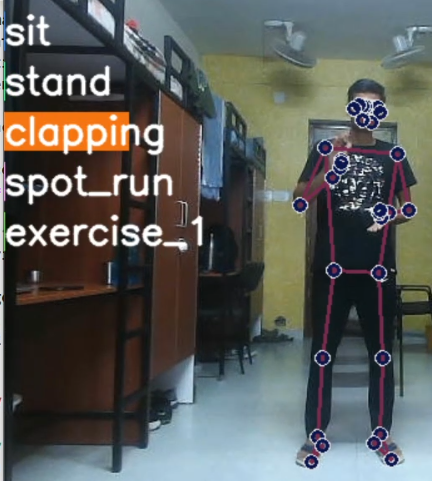
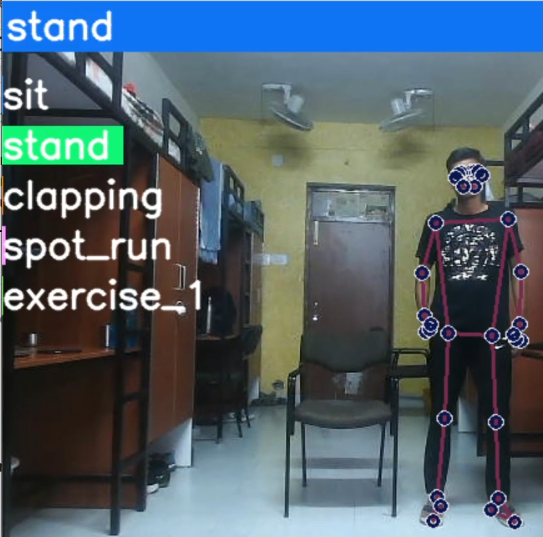
Thus, this was the implementation of the proposed method.

Fig 14 Real time Classification Fig 15 Real time Classification

**PERFORMANCE ANALYSIS**

**5.1 Performance Analysis**

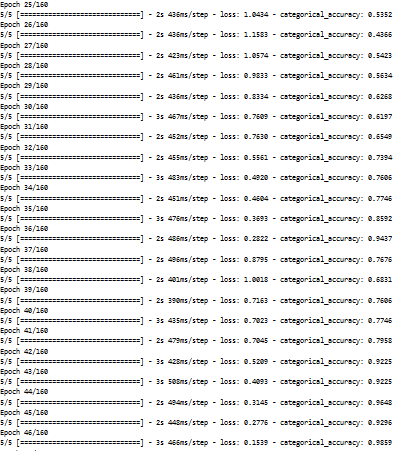
The model tests its accuracy on the training data, and is evaluated in real time.

Table 1 Accuracy and loss of model based on training data

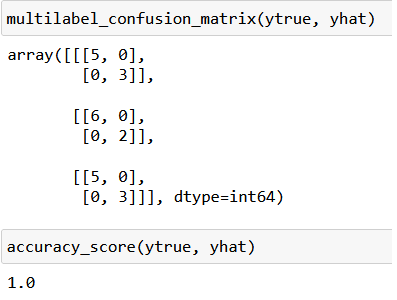


Table 2 Classification Accuracy\_score 100% on Training data

This is not a very precise metric however the Model performs very efficiently in real time so the accuracy is in acceptable ranges.

**CONCLUSION AND FUTURE WORK**

The model could be expanded to include hundreds of classes (actions) and the volume of the training data can be increased to deploy the model for general applications.

Thus, we built 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.

|  |  |
| --- | --- |
| **Applications** | **Description** |
| Virtual Reality | Promising technology that can be applied in both education and entertainment. Estimation of human posture can further clarify the relationship between the social and virtual reality world and enhance the interactive experience |
| Video Surveillance | One of the early applications use the technology in tracking, action recognition, re-identification of people within a specific range |
| Movies & Animation | Generation of various vivid digital characters is inseparable from the capture of human movements. A cheap and accurate social motion capture system can better promote the digital entertainment development industry |
| Human–Computer Interaction | It is very important for computers and robots better to understand people’s identification, location, and action. With humans’ posture, computers and robots can efficiently execute instructions and be more intelligent |
| Self-Driving | Advanced self-driving cars can respond more appropriately to pedestrians and offer more comprehensive interaction with traffic coordinators |
| Medical Assistance | It can provide physicians with quantitative human motion information, especially for rehabilitation training and physical therapy |
| Sports Motion Analysis | Estimating players’ posture in sports videos can further obtain the statistics of athletes’ indicators. It can provide a quantitative analysis of action details. |

Table 3 Applications

**REFERENCES**

1. Güler, R.A.; Neverova, N.; Kokkinos, I. DensePose: Dense Human Pose Estimation in the Wild. arXiv 2018, arXiv:1802.00434. 22. Fang, H.-S.; Xie, S.; Tai, Y.-W.; Lu, C. RMPE: Regional Multi-Person Pose Estimation. In Proceedings of the 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), Venice, Italy, 22–29 October 2017; pp. 2334–2343.

2. 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 A Comprehensive Survey of Vision-Based Human Action Recognition Methods 1,2 Department of Computer Science and Technology, Huaqiao University, Xiamen 361000, China (2019)

3. Simonyan, K.; Zisserman, A. Two-Stream Convolutional Networks for Action Recognition in Videos. In Proceedings of the NIPS 2014: Neural Information Processing Systems Conference, Montreal, QC, Canada, 8–13 December 2014; pp. 568–576.

4. Tran, D.; Bourdev, L.D.; Fergus, R.; Torresani, L.; Paluri, M. Learning Spatiotemporal Features with 3D Convolutional Networks. In Proceedings of the IEEE International Conference on Computer Vision 2015, Las Condes, Chile, 11–18 December 2015; pp. 4489–4497.

5. Ziaeefard, M.; Bergevin, R. Semantic human activity recognition: A literature review. Pattern Recognit. 2015, 48, 2329–2345.

**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

• 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 ClimentPérez 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?v=brwgBf6VB0I)

• Human Pose Estimation using opencv | python | OpenPose | stepwise implementation for beginners - YouTube (<https://www.youtube.com/watch?v=9jQGsUidKHs>)

• Review: DeepPose — Cascade of CNN (Human Pose Estimation) | by Sik-Ho Tsang | Towards Data Science ([https://towardsdatascience.com/review deeppose-cascade-of-cnn-human-pose-estimation-cf3170103e36](https://towardsdatascience.com/review%20deeppose-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.com/blog/2021/01/imageclassification-using-convolutional-neural-networks-a-step-bystep-guide/)

• Introduction to MediaPipe | LearnOpenCV (https://learnopencv.com/introductionto-mediapipe/)

• https://opencv.org/