DETECTING SHAKING HEAD IN YOUTUBE VIDEOS: A DEEP LEARNING APPROACH

An Undergraduate Research Scholars Thesis

by

YINGTAO JIANG

Submitted to the LAUNCH: Undergraduate Research office at
Texas A&M University
in partial fulfillment of the requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by
Faculty Research Advisor: Dr. Anxiao Jiang

May 2021

Major: Computer Engineering, Computer Science Track

TABLE OF CONTENTS

Page

ABSTRACT

Detecting Nodding in YouTube Videos: a Deep Learning Approach

Yingtao Jiang
Department of Computer Science and Engineering
Texas A&M University

Research Faculty Advisor: Dr. Anxiao Jiang
Department of Computer Science and Engineering
Texas A&M University

This thesis discusses the project of creating a human action detector to YouTube videos by deep learning and other tools and what is expected to be done in the future. The human action specifically studied is "shaking head", which was the focus of the detector. Most of the work is done by using existing ideas to make things work. In the late of the research, we also introduced OpenPose body landmarks to try to improve the efficiency of the model. In general, a detector is built and tested. There were 550 videos been detected and more than 5,000 moments were found. However, the accuracy needs to be further improved. The false-positive rate is 41.2%, while the false-negative rate is 10.4%. Our detection algorithm has the potential to detect 336 YouTube videos with 200 to 300 seconds in 1 hour. The detection algorithm simultaneously detects the video right after it is clipped, and it does not need to download the videos, which saves a lot of time.

DEDICATION

To the people who wear mask in this difficult time

ACKNOWLEDGMENTS

Contributors

I would like to thank my faculty advisor, Dr. Anxiao Jiang, for his guidance and support throughout the course of this research.

Thanks also go to my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a memorable experience.

Finally, thanks to my family for their encouragement and understanding, and especially thanks for their funding and the heritage my grandfather left for me.

The STAIR Actions Datasets analyzed for predicting shaking head were created by STAIR Lab, Chiba Institute of Technology. STAIR Actions were published in 2018.

The OpenPose used to detect human body and facial keypoints were created by Ginés Hidalgo, Zhe Cao, Tomas Simon, Shih-En Wei, Yaadhav Raaj, Hanbyul Joo, and Yaser Sheikh. OpenPose was published in 2019.

The iLab website to visualize the result data is the work by iLab, Texas A&M University. It was only able to be viewed within Texas A&M University and it was able to be viewed from 2020.

YouTube is a trending online video platform that is used in the thesis to test the effectiveness of the detection model.

All other work conducted for the thesis was completed by the student independently.

Funding Sources

Undergraduate research was not funded by Texas A&M University in any way. I received no funding in this research, but self-funded.

1. INTRODUCTION

This project is to design and implement a method to detect shaking head as a human action through YouTube videos. It contains two parts. The first part is to train an effective deep learning model to detect a human action, which is shaking head here. The second part is to implement the deep learning model to detect the same human action in the YouTube videos.

1.1 Motivation

We, humans, nod head for agreement and shake head for disagreement. Examining shaking head as a human behavior could have broad usage in human-computer interaction. Action recognition could also be used in the human-robot interactions, as showed in the paper by Akkaladevi and Heindl [1]. They tried to make the action recognition to be applied to industrial applications. Their results showed that there is a great potential of actually implement action recognition in a real situation. Nodding and shaking can represent the basic form of determination, which are positive and negative views. As concluded by Sharma [2], "head movements coordinate speech production, regulate turn-taking, serve as back-channeling functions and can convey attitudinal and emotional information." Nodding and shaking heads together as a pair of body signs can be used to determine the attitudes of the human sitting in front of the camera. In YouTube videos, a detection method of shaking head can be used to analyze the responsive behaviors of humans. This thesis mainly studies how to recognize shaking heads in videos by deep learning methods. Another importance and challenge of this research is that YouTube videos provide a complex testing environment to study the practical implementation of neural network models from laboratories to real life. Testing in the YouTube environment could eventually lead to a more practical detection model.

We also go through the process of using OpenPose Facial and Body Landmarks to detect human actions [3, 4, 5, 6]. The motivation here is to examine the effectiveness and the representativeness of using skeletal information of humans to recognize human behaviors in videos.

1.2 Research Roadmap

This research was done in several steps. First, we prepared the data. We downloaded the STAIR Actions dataset [7] with specific groups of behaviors. The data contains thousands of small video clips. They were then trimmed into fixed lengths and add to a big multi-dimensional matrix. Then, we designed the Conv+LSTM model and train the data. After this, we tried to improve the accuracy. By doing this, we change the model and its parameters. As the result, the training accuracy reached 95% and the testing accuracy is 73%. In the next step, we deployed this model to detect YouTube videos. YouTube videos are cast in the same matrix format as the training data. It is then found that it will trigger out of memory (OOM) error when we pass more than 5 video clips in the model. At the same time, we output the result in a JSON file. After testing the deployment worked. We tried to find as many related YouTube videos as we have. We found those videos by searching keywords, or using the video list provided in the Stair Actions [7] in the automated algorithms we wrote. We detected more than 500 YouTube videos and found more than 5000 moments that are detected as "shaking head".

So far, we had built and deployed our model to detect shaking head behavior in YouTube videos. We then examined the quality of the detection and trying to improve the quality of the detection. In an effort to improve the testing accuracy, we used OpenPose face and body landmarks detection [3, 4, 5, 6]. By doing so, we used a similar approach to prepare the data and train a similar model. As the result, the model's testing accuracy reached 78%.

There are difficulties throughout the whole process. For example, the dataset is hard to be prepared. To get the required portion of the dataset in the STAIR Actions [7], we made changes to the download commands. Then we mix several categories of actions to build the comparison dataset. These videos in the dataset are then cast into the matrix form. In the training of the model, we usually find that the computing resource we have cannot support running the model we have because that the dataset is too large. At first, we tried to make our model fit the computing resource we have. Then, in order to improve the accuracy, we fund ourselves. Potentially because the input matrix is in high dimensions while the output matrix is only double dimensions, the model can be easily triggered to have Out Of Memory (OOM) error when we test the model by passing more than 5 video clips in the trained model. We then used an algorithm to have a temporary solution. In our detection of "shaking head" human action in YouTube videos, we find some phenomenons that interestingly affect the correctness of the model. The YouTube set limits to the massive download videos tools to itself, which gave us some problems in detecting behaviors in too many YouTube videos. We are lucky to find a way to get the YouTube video data without downloading it. Still, we want to improve the effectiveness and correctness of our detection to the "shaking head", which we mentioned in the last section about what are the future works to this study.

1.3 Related Works

A tremendous amount of research has been done before. Here, some of those researches and works were concluded. They are very helpful through the research process of this thesis.

1.3.1 Works on the Machine Learning Models

There are works on detecting shaking heads or various head gestures. Sharma's research revealed the variance of head gestures that make the detection difficult. To better approach the problem, he proposed a Multi-Scale Deep Convolution-LSTM (long short-term memory) architecture to recognize different head gestures, which is better than simple LSTM [2]. The combination of CNN and RNN is used in his research. A more specific explanation is that the authors used CNN to process several frames of videos first, then he connected those frames in RNN, specifically LTSM to find the mutual connected relationships. In Langholz's research, he just used RNN architecture with LSTM. [8] He claimed to reach a 91.78 % of accuracy to separate nodding, shaking head, and others. This is one of the few papers that studied shaking head. Besides doing the model training, he also used his model to the on-device prediction, which provides some inner thoughts about the real deployment of human activity detection models. He mentioned 2 things that need to be improved in the real deployment: the first thing is that the prediction will only start after collecting enough videos times, and the second thing is that there could be many head gestures happen in 4 seconds but it is only designed to predict in every 4 seconds buffer [8].

There are works on the detection of human actions. Action detection could be used in healthcare to monitoring the behavior of elderly people. For example, using the camera at home to detect accidental falls of elders, which could thus shorten the time of finding that the elders had fallen. Gao and others, in their paper, have demonstrated the effectiveness of such usage in intelligent healthcare [9]. They develop a deep learning architecture called recurrent 3D convolutional neural network (R3D). Recognizing head gesture could find some insights in recognizing hand gesture. In Lin and others' research on recognizing hand gestures, he uses a convolutional neural network (CNN) method to recognize hand gestures [10]. Their average recognition accuracy reached about 96%. This research also tries to solve the problem of skin color differences of human hands, which they used a Gaussian Mixture Model (GMM). CNN + LSTM networks are widely used in the researches in action recognition. For example, in the paper by Xi Ouyang and others, they use CNN + LSTM to combine with multi-task learning mechanism to do action recognition, thus making an efficient model for detecting video clips in different categories [11]. In the paper by Ullah and others, they also used CNN and LSTM to do action detection, but they also provide a thought that LSTM can have two directions: forward and backward [12]. They used both directions of LSTM in their model and it showed that the results are being significantly improved.

1.3.2 Works on the Related Datasets

The STAIR-Actions dataset, developed by Dr. Yoshikawa and others, has more than 100 categories of human actions [7]. Each category has about 1,000 short video clips. This action provides a dataset of shaking head which consists of two parts: the original one created by Dr. Yoshikawa, etc., and a list of YouTube videos. Each clip is about 3 to 5 seconds long, consisting of a person doing the specific actions. In the list of the YouTube videos, they also include the timestamps of when the action happened. In our research, we used the shaking-head data set of the STAIR-Action dataset and some videos clips from various categories to be the comparison group.

1.3.3 Works on the Pre-processing Tools

Through investigating on improving the detection of shaking heads, OpenPose, a real-time human body and facial keypoints detection system developed by Hidalgo, is found to be helpful [3, 4, 5, 6]. By using OpenPose to process the videos dataset, it generates a series of body landmarks that represents the body movement of humans in the videos frame by frame. Using this system has the potentials to decrease the computing resource needed to train or predict the model and improve the overall correct rate.

A similar approach of using extracted skeletal information to detect human action is proposed by Mathe and others. , They extracted the data of human joints from a complex transformation. They also examined their idea by using their approach in a dataset of human actions [13]. This paper showed that skeletal information could be used to detect human actions. Not only can we extract skeletal information, but we could also extract the motion information in videos. In the paper by Lu Xia and others, they used 3D joint locations to do action recognition [14]. Their method in using skeletal information reached a prominent result. In Wang and others' paper, they propose a method that getting the information of motion and then using an attention model to do action recognition [15]. Their result showed that this approach is useful.

We also need to intensively cast the videos, where we use the OpenCV library [16]. OpenCV provides fast caption tools to access the YouTube videos without downloading them. It was also used in casting the videos into consistent clips.

2. METHODS

The methods used in this research are not unique, but are the combination of many previous works to make a full detection method of shaking head in YouTube.

2.1 Proposed Deep Learning Model

Based on the Multi-Scale Deep Convolution-LSTM architecture by Sharma [2], we proposed a CONV2+LSTM model. The model start with Conv2D, then it is flattened to the LSTM and followed by dense some layers, as (**Fig. 2.1**) shows.

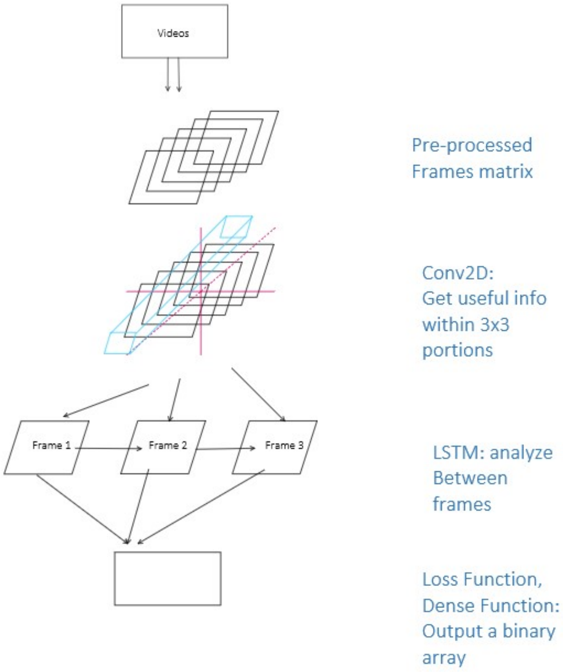
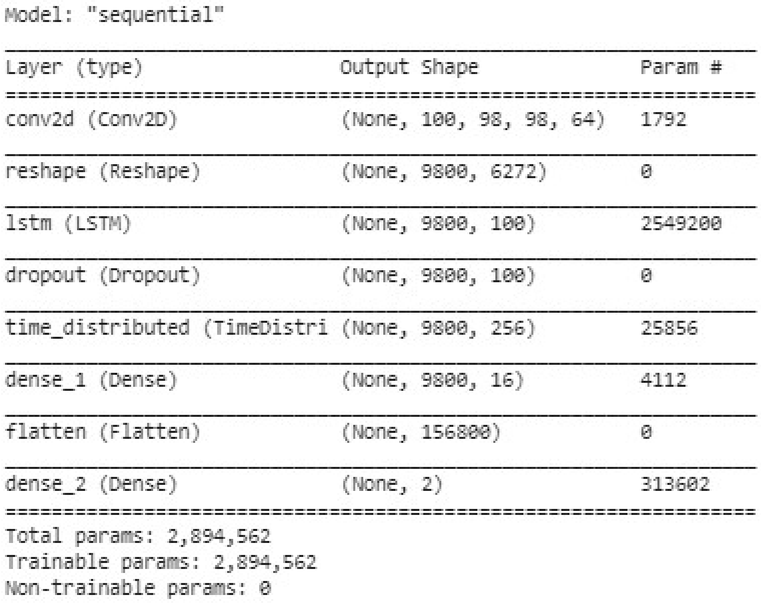


Figure 2.1: Conv2D+LSTM model design

The data is first analyzed by the Conv2D layer, where it divides each frame into 3x3 portions and analyzes within them. Then, it was passed to the LSTM, which is responsible to analyze the relationship between different frames. At last, it will output a binary double array which the index

0 within it is the probability of the clip of the video that should not be "shaking head", while index 1 is the probability of the clip of the video that should be "shaking head". The input tensor of this model is [100, 100, 100, 3], where each of the 100s represents the parameters of a clip, which is composed of 100 frames with image pixels 100 times 100, and there are 3 channels. The output shape is a 2D array with [number of clips, 2], which consists of the prediction for each video clip with an array where the probability that is "shaking head" is at position 0 while not at position 1. The output tensor can be found clearly in (**Fig. 2.2**).

This model makes sense because when the action happens, which is "shaking head" here, we want to analyze a part of the video frame that may contain necessary movement, and we want to analyze the relationship between frames as an action is done in a range of times, which means that the action is done in several frames.



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 100, 98, 98, 64)   1792
_________________________________________________________________
reshape (Reshape)            (None, 9800, 6272)        0
_________________________________________________________________
lstm (LSTM)                  (None, 9800, 100)         2549200
_________________________________________________________________
dropout (Dropout)            (None, 9800, 100)         0
_________________________________________________________________
time_distributed (TimeDistri (None, 9800, 256)         25856
_________________________________________________________________
dense_1 (Dense)              (None, 9800, 16)          4112
_________________________________________________________________
flatten (Flatten)            (None, 156800)            0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 313602
=================================================================
Total params: 2,894,562
Trainable params: 2,894,562
Non-trainable params: 0
```

Figure 2.2: Parameters of the Model

2.2 Using Existing Dataset

The STAIR-Actions dataset [7] contains a database of videos that originated from the developers of STAIR-Actions and a YouTube videos ID list for each category of actions it has. The database is used in training the model, while the YouTube ID list is used for detection and a part of the shaking head database developed by this study.

As different from other action detection works that detect multiple actions, in this study, the detection is to determine whether it has a "shaking head" in a clip of videos, which is a binary classification process. On the one hand, I already had a data set of "shaking head", on the other hand, I had to prepare a data set that is "not-shaking head". To achieve this, I randomly populated videos in other categories of actions and build this "not-shaking head" data set. The "shaking head" data set and "not-shaking head" have a ratio of about 1:2 and the total numbers of videos are about 1800.

After getting the data sets, they were processed into a matrix. Each video, which is about 3 to 7 seconds long with 30 frames per second, was cast into an array with 100 frames, 100 heights, and 100 lengths, and were connected together to create matrix X, which contains videos clips, and matrix Y to represent which type they are. Then X and Y would be used in the deep learning model.

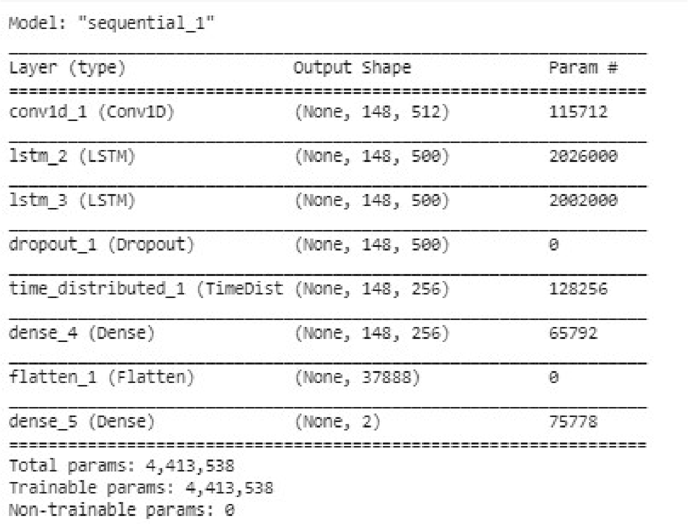
The list of YouTube videos was extracted from the STAIR Actions [7] by sorting the videos that claim to have "shaking head" in them, which is used to construct the YouTube database.

2.3 Using OpenPose to Refine the Detection Model

The deep learning model proposed before does not make a testing correct rate higher than 80%. Then, OpenPose is proposed to improve the correct rate [3, 4, 5, 6]. In the deep learning model proposed previously, it is a one-step process with a video input and a binary output. However, the introduction of OpenPose made the detection of human action in videos into two steps. The first step is to find the exact position of humans and their body landmarks in the videos, and the second is to find the relation of shaking behavior with those human body landmarks. This proposed a more systematic way of detection.

OpenPose generates the body landmarks of the people in the video dataset, which contains 25 key points, 3 values per keypoints. Here, we assume that there will only be one person in one frame of a video. For the frame that was detected no data by OpenPose occasionally, we suppose that the body keypoint landmarks are all 0 for that frame. Provided that there are 150 frames per video, which is about 5 seconds, a new matrix X is created. A similar CNN + LSTM model will be used to train the model.

The strength of using OpenPose to process the video data is that it provides meaningful information contained in the videos. It is also a tool to find people in the videos. Thus we could use this quality in YouTube videos to make sorting that excludes video clips does not contain people in it. Using OpenPose saves computing resources need to run the model since its data is only 75 float values per frame, instead of the original video input that is a 2D-array with 100 $\times 100$ per frame. The model is as you could see in the (**Fig. 2.3**),



```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv1d_1 (Conv1D)            (None, 148, 512)          115712    
_________________________________________________________________
lstm_2 (LSTM)                (None, 148, 500)          2026000   
_________________________________________________________________
lstm_3 (LSTM)                (None, 148, 500)          2002000   
_________________________________________________________________
dropout_1 (Dropout)          (None, 148, 500)          0         
_________________________________________________________________
time_distributed_1 (TimeDist (None, 148, 256)          128256    
_________________________________________________________________
dense_4 (Dense)              (None, 148, 256)          65792     
_________________________________________________________________
flatten_1 (Flatten)          (None, 37888)             0         
_________________________________________________________________
dense_5 (Dense)              (None, 2)                 75778     
=================================================================
Total params: 4,413,538
Trainable params: 4,413,538
Non-trainable params: 0
```

Figure 2.3: Parameters of the Model Using OpenPose

The methodology of using OpenPose is to make the future work of making stronger action detection tools possible.

2.4 Detection of Shaking Head in YouTube Videos

As you could see in (**Fig. 2.4**), the method of detecting action in the YouTube videos is as following: First, it will load a YouTube video by a YouTube ID. Then, it will be cut and resize at the same time into the same format as the training dataset, which means that the video is resized and cut into several video clips about 3.33 seconds. After that, it is safe to be passed into the trained deep learning model we got.

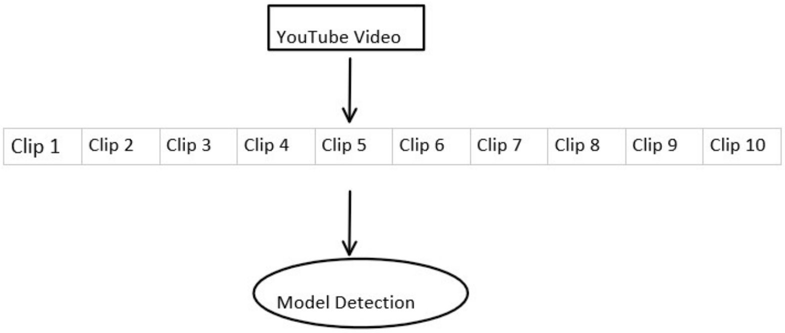


Figure 2.4: YouTube Detection Process

Given a YouTube video, which means that the YouTube ID was given to the program, the video is divided into pieces of 100 frames each with pafy and OpenCV libraries, which is stable than sole OpenCV [16]. 100 frames mean that each video clip is 3.33 seconds long. Then, they were examined individually by using the deep learning model. Because the deep learning model will trigger Out of Memory (OOM) error when detecting too many clips, each detection is set to only detect 5 clips, and a video needs detection to be done several times. Because the video does not need to be download and the detection is done simultaneously, the time spent on detecting a set of videos is not too long.

No fancy way was used except doing detection simultaneously, but this way it makes sure that each piece of video was detected. "Shaking head" has its uniqueness in occurrence that it can occur anywhere and the title of the YouTube videos cannot make reasonable hint as there is this behavior in the videos (If searching "shaking head" in YouTube, the videos it showed are in small quantities and cannot be used), but we still used related words detection or frames extraction here.

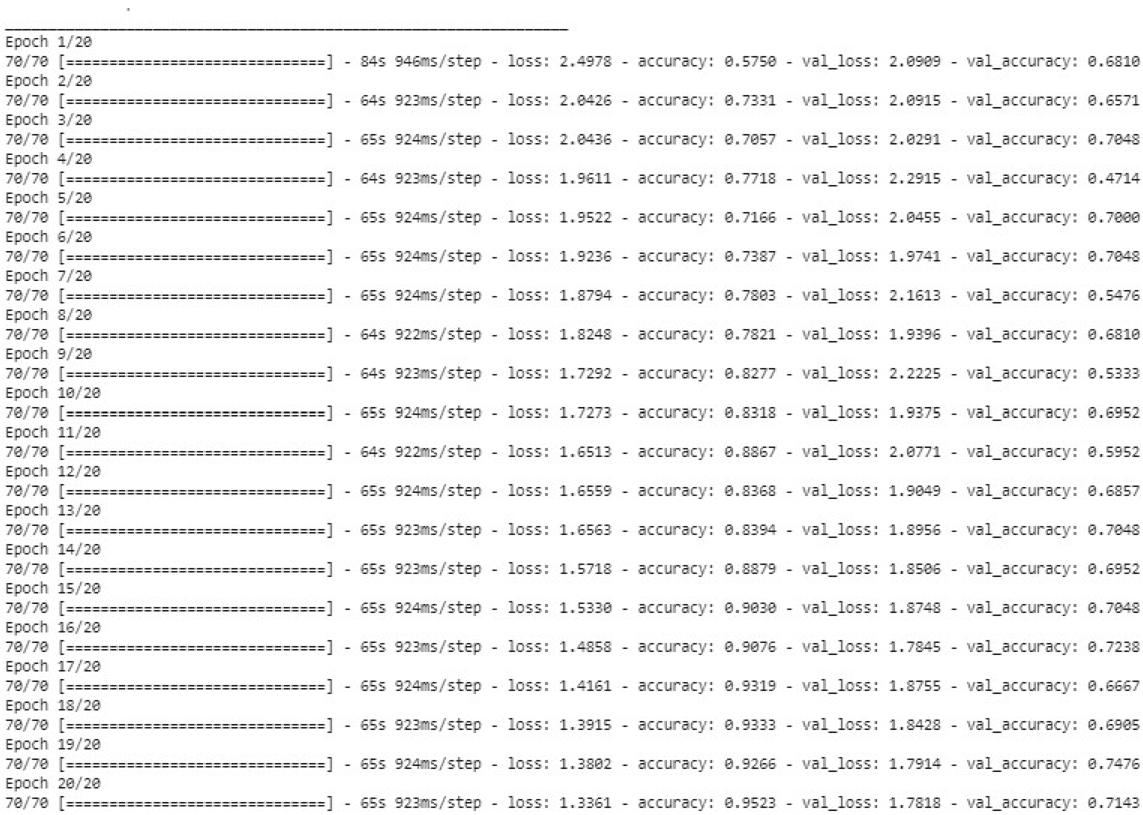
The output of the detection to the YouTube videos is in the JSON format, where we stored necessary information of the video clips that are detected to contain "shaking head" action. It records download the start time and the end time of the action, which the duration is a fixed time in 3.33 seconds. It also contains information like who detects this, which human action it is defined, and the expected correct rate for this detection. This JSON file can be used to track the detection and it could be easily imported into the database.

3. RESULTS

The training model and detection model are tested and the results are as following.

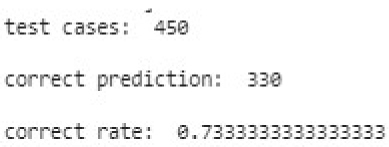
3.1 Performance of the Proposed Model

There is in total 5 layers besides "Reshape" layer, "Flatten" layer, etc. The first layer, Conv2D, has filter 64 and its kernel size is (3, 3). The next layer, LSTM, has a filter of 100. Then it is followed by 3 dense layers that its units decrease gradually as this is found helpful in increasing accuracy. As shown in (**Fig. 3.1**), after training in 20 epochs, the training set accuracy reaches 0.95 and the validation set accuracy reaches 0.71. In (**Fig. 3.1**), we could see that there is overfitting after epoch 15, but the validation set accuracy, after overfitting, increases back to the same rate as before the overfitting. Then at epoch 20, the training accuracy is 0.95 while validation accuracy reaches 0.71, which constitutes an underfitting. I then test the accuracy of the model on the testing set, as in (**Fig. 3.2**), which reaches an accuracy of about 0.73, which matches with the validation accuracy. Here, the testing set is consists of 30% of the whole data set, while the training set and validation set split the rest data set as 4 to 1.



```
Epoch 1/20
70/70 [==============================] - 84s 946ms/step - loss: 2.4978 - accuracy: 0.5750 - val_loss: 2.0909 - val_accuracy: 0.6810
Epoch 2/20
70/70 [==============================] - 64s 923ms/step - loss: 2.0426 - accuracy: 0.7331 - val_loss: 2.0915 - val_accuracy: 0.6571
Epoch 3/20
70/70 [==============================] - 65s 924ms/step - loss: 2.0436 - accuracy: 0.7057 - val_loss: 2.0291 - val_accuracy: 0.7048
Epoch 4/20
70/70 [==============================] - 64s 923ms/step - loss: 1.9611 - accuracy: 0.7718 - val_loss: 2.2915 - val_accuracy: 0.4714
Epoch 5/20
70/70 [==============================] - 65s 924ms/step - loss: 1.9522 - accuracy: 0.7166 - val_loss: 2.0455 - val_accuracy: 0.7000
Epoch 6/20
70/70 [==============================] - 65s 924ms/step - loss: 1.9236 - accuracy: 0.7387 - val_loss: 1.9741 - val_accuracy: 0.7048
Epoch 7/20
70/70 [==============================] - 65s 924ms/step - loss: 1.8794 - accuracy: 0.7803 - val_loss: 2.1613 - val_accuracy: 0.5476
Epoch 8/20
70/70 [==============================] - 64s 922ms/step - loss: 1.8248 - accuracy: 0.7821 - val_loss: 1.9396 - val_accuracy: 0.6810
Epoch 9/20
70/70 [==============================] - 64s 923ms/step - loss: 1.7292 - accuracy: 0.8277 - val_loss: 2.2225 - val_accuracy: 0.5333
Epoch 10/20
70/70 [==============================] - 65s 924ms/step - loss: 1.7273 - accuracy: 0.8318 - val_loss: 1.9375 - val_accuracy: 0.6952
Epoch 11/20
70/70 [==============================] - 64s 922ms/step - loss: 1.6513 - accuracy: 0.8867 - val_loss: 2.0771 - val_accuracy: 0.5952
Epoch 12/20
70/70 [==============================] - 65s 924ms/step - loss: 1.6559 - accuracy: 0.8368 - val_loss: 1.9049 - val_accuracy: 0.6857
Epoch 13/20
70/70 [==============================] - 65s 923ms/step - loss: 1.6563 - accuracy: 0.8394 - val_loss: 1.8956 - val_accuracy: 0.7048
Epoch 14/20
70/70 [==============================] - 65s 923ms/step - loss: 1.5718 - accuracy: 0.8879 - val_loss: 1.8506 - val_accuracy: 0.6952
Epoch 15/20
70/70 [==============================] - 65s 924ms/step - loss: 1.5330 - accuracy: 0.9030 - val_loss: 1.8748 - val_accuracy: 0.7048
Epoch 16/20
70/70 [==============================] - 65s 923ms/step - loss: 1.4858 - accuracy: 0.9076 - val_loss: 1.7845 - val_accuracy: 0.7238
Epoch 17/20
70/70 [==============================] - 65s 924ms/step - loss: 1.4161 - accuracy: 0.9319 - val_loss: 1.8755 - val_accuracy: 0.6667
Epoch 18/20
70/70 [==============================] - 65s 923ms/step - loss: 1.3915 - accuracy: 0.9333 - val_loss: 1.8428 - val_accuracy: 0.6905
Epoch 19/20
70/70 [==============================] - 65s 924ms/step - loss: 1.3802 - accuracy: 0.9266 - val_loss: 1.7914 - val_accuracy: 0.7476
Epoch 20/20
70/70 [==============================] - 65s 923ms/step - loss: 1.3361 - accuracy: 0.9523 - val_loss: 1.7818 - val_accuracy: 0.7143
```

Figure 3.1: Validation Accuracy of the Model



```
test cases:  450

correct prediction:  330

correct rate:  0.7333333333333333
```

Figure 3.2: Testing Accuracy of the Model

There are many different layers and parameters that were tested. Many different layers were tried such as Conv3d, ConvLSTM2D, but they cannot improve the performance of the model. Also, I tried to add more layers of Conv2D and LSTM to the model but adding layers cannot improve the performance, too. I use the Dropout() layer to reduce overfitting.

This model could do jobs to determine the "shaking head", as it shows the ability to find this behavior. However, obviously the better the accuracy the better its real performance. This model has the potential to be improved by dealing with overfitting and underfitting, but considering the limitation of this model, the current result is very close to the expectation of this model.

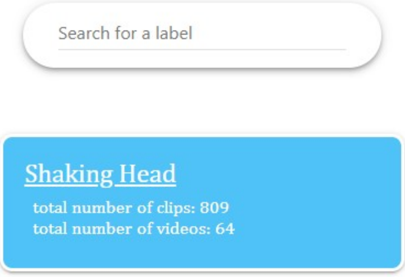
This dataset has biases and flaws. The data set's samples are all Asians in the Asian settings. Though this dataset shows possible circumstances of "shaking head", it cannot show differences in the diversity of human races and cultures. This could greatly affect the later detection in YouTube Videos. The model cannot do well detection when the resolution of the original video clips is low; there is more than 1 person in frames; or there is some tricks in the video clip. The model can do well when there is only 1 person in the frames and the action happen entirely in the center of the video clip.

The model performance is not as satisfactory as originally thought. It is potentially due to that the data of the comparison group is not good; the data of "shaking head" does not show enough attributes to such behavior; binary classification was used instead of multiple classifications, which parameters were sharply dense into 2 choices instead of multiple choices with possible lots of loss or under-representation of many important parameters in the matrix.

3.2 Performance on Detecting Moments in YouTube Videos

3.2.1 Result of Mass Detection to YouTube Videos

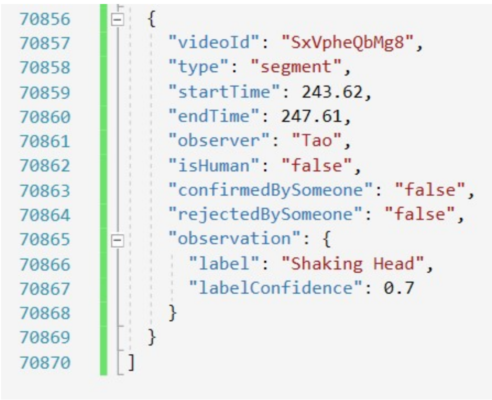
You could visualize the data on the website of iLab. The label is "Shaking Head" on the iLab website to visualize the data. As shown in (**Fig. 3.3**) below, there are now 64 videos and more than 800 moments can be viewed on the iLab website. to view them, just go to the iLab website and choose "Shaking Head". Each video has time stamps which are moments that are with fixed length of 3.33 seconds (100 frames). There are also 550 videos with about 5000 moments waiting to be uploaded, as in (**Fig. 3.4**). Because the JSON file is too large (with more than 80,000 lines) so it cannot be all uploaded to the website.



Shaking Head

total number of clips: 809
total number of videos: 64

Figure 3.3: iLab database "shaking head"



```
  {
    "videoId": "SxVpheQbMg8",
    "type": "segment",
    "startTime": 243.62,
    "endTime": 247.61,
    "observer": "Tao",
    "isHuman": "false",
    "confirmedBySomeone": "false",
    "rejectedBySomeone": "false",
    "observation": {
      "label": "Shaking Head",
      "labelConfidence": 0.7
    }
  }
]
```

Figure 3.4: JSON file, result of detecting on YouTube videos

3.2.2 Performance and Accuracy of the Detection

To examine the accuracy of the detection, we use false-positive rate (FPR). Here, "shaking head" is defined as an action of the angular movement of the head and the FPR as the percentage of obviously false detection in all the moments that were detected as true. The first 20 videos of "shaking head" in the iLab database were tested and this test is done manually. For the 250 clips detected as "shaking head", 147 of them are proved as "shaking head" by giving some tolerance for some actions that are not perfectly "shaking head". The FPR rate then got to be 41.2%. Common things that lead the model to detect wrongly are that head is presenting but not shaking; more than

1 person is in the video; and graphs of heads are treated as real heads by the model.

False-negative rate (FNR) is also determined by random selection of 5 videos in the videos. It is taken as:

$$\frac{N_{notdetected}}{N_{notdetected} + N_{detected}} \qquad \text{(Eq. 1)}$$

where "detected" means all moments that the detection model thought to be "shaking head". For the 5 videos, 10 moments were newly founded manually, with 39 moments found by the computer, so the FNR is 20.4%.

In the process of determining FPR and FNR, we found that some videos were detected better than others. Some videos were detected almost all correct while some videos have a large quantity of false detection. In general, the FPR and FNR are satisfactory considering the accuracy of the model. Though FPR is relatively high, does this mean that the detection is completely insufficient? No, the detection did jobs greatly in providing a set of moments that is very close to the "shaking head" action. To improve the FPR and FNR, we need to either improve the model or change the training data since the training data has a huge difference from the YouTube videos we tested here.

3.3 Improve Accuracy and Efficiency of the Detection

3.3.1 Improve Accuracy of Model on YouTube Videos

As mentioned earlier, efforts were made by tried to change the model by adding layers or change layers but they were not strongly helpful. Then we do not believe that by simply changing the model, the accuracy of the model can be further improved. There are distinctions for the training data set that they have no relations with the YouTube videos they are going to detect, and the comparison data, "not shaking head", is not a complete dataset which was just coming from 10 other data set of different categories. However, if we only care about FPR, the accuracy that the detected moments has the action we want, a simple way is to take moments that the prediction model is more confident about, instead of taking every moment that has a higher probability to be the action than not. By not accepting moments that the prediction model did not predict with higher probability, for example, 0.7 could be used, the FPR can then be increased.

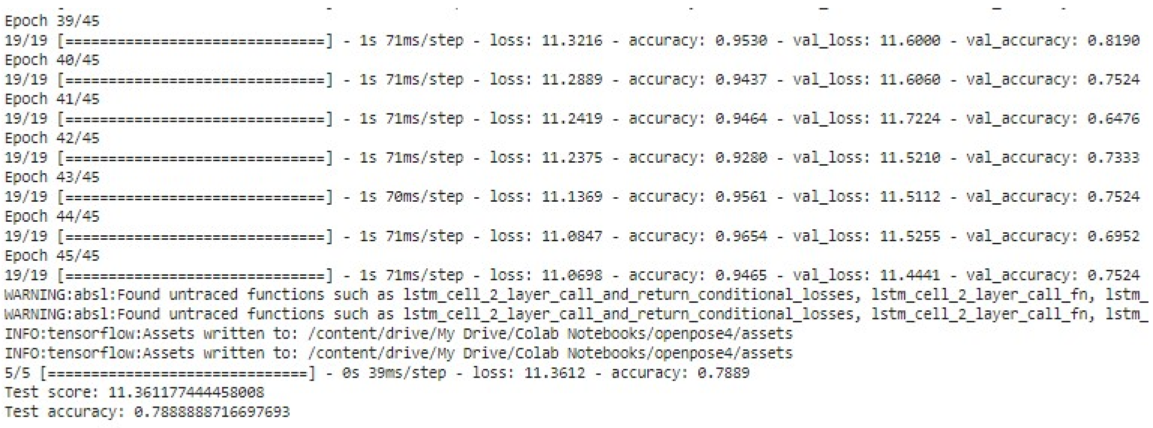
3.3.2 Improve Efficiency of Shaking Head Detection

To improve the efficiency of action and emotion detection, several query tools are made to find actions. The STAIR Actions dataset provides a list of YouTube videos. These videos were detected first. Also, a tool is developed that could search on keywords of the videos' titles. It can detect the videos that contain certain keywords in the titles, like "shaking head", "shaking" or others. We assume that there is a correlation between its title and its content, so that query on titles could help find some videos that are more possible to have "shaking head" behavior than other videos.

3.3.3 Result of Using OpenPose in Model Training

The final deep-learning model of using OpenPose JSON data is mostly like the original model. It comes up with a 1 dimension convolution neural network (CNN) and followed by several layers of long short term memory (LSTM). We did not used the video data that OpenPose marks on each frame for each body landmark. We thought that directly using the location data of those body landmarks could be effective in determine the human action. It makes sense because that analyzing the relative positions of human in a consecutive frames are how we, humans, determine the behaviors in the video. Hence, the deep learning model could also finding the attributes in the changes of body landmarks over times.

The training time of using the processed video data by OpenPose is much faster than using the complex matrix that contains detailed information of each frame, though using OpenPose to process the video cost some time. We also tried different parameters to try to reach a maxmized result. The result training accuracy is 0.9465 and the testing accuracy is 0.7889 after 45 epochs. The whole training process costs in about 2 minutes. Though it does not reach the much higher score of testing accuracy as expected, it tremendously improves the efficiency in training or using the model to predict, as in (**Fig. 3.5**). We still need to find a way to shorten the time using the OpenPose to get the human body landmarks. By doing this, the efficiency could be further improved.



```
Epoch 39/45
19/19 [==============================] - 1s 71ms/step - loss: 11.3216 - accuracy: 0.9530 - val_loss: 11.6000 - val_accuracy: 0.8190
Epoch 40/45
19/19 [==============================] - 1s 71ms/step - loss: 11.2889 - accuracy: 0.9437 - val_loss: 11.6060 - val_accuracy: 0.7524
Epoch 41/45
19/19 [==============================] - 1s 71ms/step - loss: 11.2419 - accuracy: 0.9464 - val_loss: 11.7224 - val_accuracy: 0.6476
Epoch 42/45
19/19 [==============================] - 1s 71ms/step - loss: 11.2375 - accuracy: 0.9280 - val_loss: 11.5210 - val_accuracy: 0.7333
Epoch 43/45
19/19 [==============================] - 1s 70ms/step - loss: 11.1369 - accuracy: 0.9561 - val_loss: 11.5112 - val_accuracy: 0.7524
Epoch 44/45
19/19 [==============================] - 1s 71ms/step - loss: 11.0847 - accuracy: 0.9654 - val_loss: 11.5255 - val_accuracy: 0.6952
Epoch 45/45
19/19 [==============================] - 1s 71ms/step - loss: 11.0698 - accuracy: 0.9465 - val_loss: 11.4441 - val_accuracy: 0.7524
WARNING:absl:Found untraced functions such as lstm_cell_2_layer_call_and_return_conditional_losses, lstm_cell_2_layer_call_fn, lstm_
WARNING:absl:Found untraced functions such as lstm_cell_2_layer_call_and_return_conditional_losses, lstm_cell_2_layer_call_fn, lstm_
INFO:tensorflow:Assets written to: /content/drive/My Drive/Colab Notebooks/openpose4/assets
INFO:tensorflow:Assets written to: /content/drive/My Drive/Colab Notebooks/openpose4/assets
5/5 [==============================] - 0s 39ms/step - loss: 11.3612 - accuracy: 0.7889
Test score: 11.361177444458008
Test accuracy: 0.7888888716697693
```

Figure 3.5: Training Result by using Openpose facial and body land mark

4. CONCLUSION

4.1 Conclusion

In this thesis, we explore the process to detect a human action in YouTube videos, which is shaking head. In the whole process, a video is first been processed into a uniform form, and cut into video clips. Then it will be detected by a trained deep learning model developed in this thesis. In the late of this research, an improvement was introduced, which is the OpenPose body landmarks [3, 4, 5, 6].

In this thesis, a deep learning model to detect shaking heads is built. The testing correct rate is about 73%. Then, a detection model to detect YouTube video is built, with an output of a JSON file that contains the prediction information. The false-positive rate, as manually measured by me, is 41.2%. By using this detection model, 550 videos were determined and more than 5,000 shaking head behaviors were found.

In general, a basic version of the action detector to detect "shaking head" within YouTube videos was built. The performance could be more satisfactory to meet the expectation, which means more improvements are needed. Some problems were faced with temporary solutions were used but may need a better solution, like the OOM when predicting the model and under-representation of the training data of "shaking head".

By using the human body landmarks of OpenPose [3, 4, 5, 6], we were able to slightly improve the accuracy of the model and also greatly improve the training time of the deep learning model. The testing accuray of the model using the OpenPose huam body landmarks is about 78%. OpenPose provides a good foundation for conducting research in human action detection. However, the problem with using OpenPose is that it costs time for using the OpenPose to generate body landmarks in the videos. There is no solution now since we use the integrated tool of OpenPose that cannot be easily modified. This is a good try in improving the efficiency of detecting actions in the YouTube videos.

4.2 Future Research

There could be three directions for future research. First, we could use a similar approach to build a model to detect other human actions. By doing this, we could develop a systematic way of detecting human actions in videos. This systematic way should be efficient, which will make it possible to be widely used in the real life. Hence, making the vision of computers more powerful. To get a more well-rounded action detection, we could train a model that uses more than one dataset. Cao's research on the action detection cross the datasets is done in order to detect action in a different dataset [17]. The approach used in Cao's research reduces the need to train labels, potentially eliminating the tedious work required of labeling.

Second, we could refine the model. Currently, the approach used in this thesis is relatively heavy and awkward. We could develop a detection tool that could output the exact location of the human action in one frame and also tell the exact start time of one action and also the end time of it. Yeung's research could be useful in finding the start and end of an action in the videos [18]. He developed a ful end-to-end approach to determine which frame action is occurring. This approach could solve the problem of determining the start and endpoint of action. Ping Wei and others, in his paper, said that multiple actions could happen in one clip of a video, which he then designed a model that could detect multiple actions [19]. His idea is profound. In the deployment of a trained model to detect in YouTube videos, we need to concern the same situation that multiple actions could happen in the same time interval. His research could be useful in developing a useful action detector. Not only should we care about when and the duration of human action, but we also should care about the location in the frame of where the action happened. One approach proposed by Peng and Schmid is using a multi-region two-stream R-CNN model for action detection [20].

Also, in the later research, we used two-step detection. The first step is using OpenPose. OpenPose actually provides the skeleton information in the videos. There could be other ways and methods to get the skeleton information and the format of it could be diverse. OpenPose, in its root, is a tool to generate body landmarks, which may not be a sufficient way to get the key information of human actions. We could work on providing a way to process videos and photos that makes the object detection accessible to be used under low computing resource. Also, OpenPose is robust but also not so efficient. In a real deployment, we should consider other tools that did the same thing but are designed to be used on mobile devices or faster.

REFERENCES

[1] S. C. Akkaladevi and C. Heindl, "Action recognition for human robot interaction in industrial applications," in *2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS)*, pp. 94–99, IEEE, 2015.

[2] M. Sharma, D. Ahmetovic, L. A. Jeni, and K. M. Kitani, "Recognizing visual signatures of spontaneous head gestures," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 400–408, IEEE, 2018.

[3] Y. Yoshikawa, J. Lin, and A. Takeuchi, "Stair actions: A video dataset of everyday home actions," *arXiv preprint arXiv:1804.04326*, 2018.

[4] T. Simon, H. Joo, I. Matthews, and Y. Sheikh, "Hand keypoint detection in single images using multiview bootstrapping," in *CVPR*, 2017.

[5] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *CVPR*, 2017.

[6] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *CVPR*, 2016.

[7] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[8] E. H. Langholz and R. Brasher, "Real-time on-device nod and shake recognition," *arXiv preprint arXiv:1806.04776*, 2018.

[9] Y. Gao, X. Xiang, N. Xiong, B. Huang, H. J. Lee, R. Alrifai, X. Jiang, and Z. Fang, "Human action monitoring for healthcare based on deep learning," *IEEE Access*, vol. 6, pp. 52277–52285, 2018.

[10] H.-I. Lin, M.-H. Hsu, and W.-K. Chen, "Human hand gesture recognition using a convolution neural network," in *2014 IEEE International Conference on Automation Science and Engineering (CASE)*, pp. 1038–1043, IEEE, 2014.

[11] X. Ouyang, S. Xu, C. Zhang, P. Zhou, Y. Yang, G. Liu, and X. Li, "A 3d-cnn and lstm based multi-task learning architecture for action recognition," *IEEE Access*, vol. 7, pp. 40757–40770, 2019.

[12] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, and S. W. Baik, "Action recognition in video sequences using deep bi-directional lstm with cnn features," *IEEE access*, vol. 6, pp. 1155–1166, 2017.

[13] E. Mathe, A. Maniatis, E. Spyrou, and P. Mylonas, "A deep learning approach for human action recognition using skeletal information," in *GeNeDis 2018*, pp. 105–114, Springer, 2020.

[14] L. Xia, C.-C. Chen, and J. K. Aggarwal, "View invariant human action recognition using histograms of 3d joints," in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. 20–27, IEEE, 2012.

[15] J. Wang, W. Wang, and W. Gao, "Fast and accurate action detection in videos with motion-centric attention model," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 1, pp. 117–130, 2018.

[16] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[17] L. Cao, Z. Liu, and T. S. Huang, "Cross-dataset action detection," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1998–2005, IEEE, 2010.

[18] S. Yeung, O. Russakovsky, G. Mori, and L. Fei-Fei, "End-to-end learning of action detection from frame glimpses in videos," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2678–2687, 2016.

[19] P. Wei, N. Zheng, Y. Zhao, and S.-C. Zhu, "Concurrent action detection with structural prediction," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3136–3143, 2013.

[20] X. Peng and C. Schmid, "Multi-region two-stream r-cnn for action detection," in *European conference on computer vision*, pp. 744–759, Springer, 2016.