Posture correction for Weight lifting

Harishanth Sivakumaran[1], Suraj Sudhakar[1], and Suhas Gopal[1]

Saarland University, Germany

Abstract. A perfect method must be followed by the weightlifters to avoid injury. We try to automate the process of detecting wrong posture using neural network models. First, we determine the key points and estimate the pose. Next, we try two different approaches, in one approach, this skeletal structure is then fed to a second CNN and in another approach we feed the keypoints to a DNN model, which will classify the posture as good or bad posture.

Keywords: Weightlifting · Human Pose Estimation · Convolution Neural Network

1 Introduction

Exercising is good for health, but if performed incorrectly it can lead to more issues than benefits. This is especially true in the case of weightlifting. To understand how human body part moves for an exercise we can use Keypoint detection to identify the pose. This can be used to classify whether an action performed is in correct posture or not.

In this project, we have considered deadlift exercise for classification.

1.1 Postures in Deadlift

Some of the common bad postures while performing deadlift is shown below.

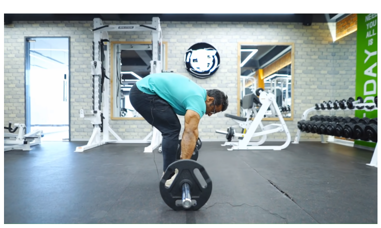

Fig. 1. Bent back

Fig. 2. Over-extending

Some of the good postures are shown below:

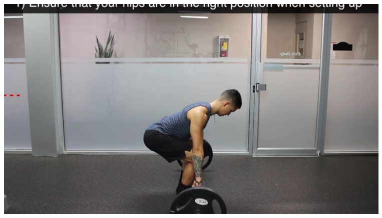

Fig. 3. Straight back posture

Fig. 4. Not over-extended

1.2 Pose Estimation

Pose estimation can be used to determine the spatial locations of key body points. This can be used on images or videos of humans to get the skeletal structure.

Fig. 5. Keypoints

Our idea is to use this skeletal image and classify the posture as good or bad posture.

2 Related Work

In the paper [1], the authors perform sitting posture classification using pose estimation based on OpenPose followed by a classification using a 19 layer model.

3 Proposed Method

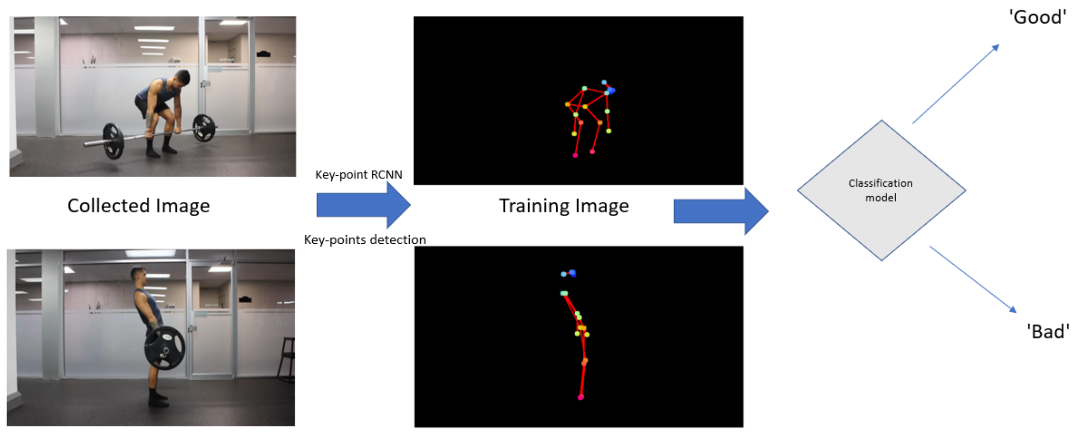

Fig. 6. Workflow

We divide the proposed method into the following three sub parts.

1. Data Collection
2. Keypoint Annotation
3. Posture classification

3.1 Data Collection

We did not find any existing dataset for the deadlift exercise to suit our purpose. So we collected the images manually from youtube videos and labelled as 'Good' or 'Bad' posture depending on the comments from the narrator of the video. Some of the issues faced during data collection phase.

- Sometimes it was difficult to get a clear screenshot as frames usually consisted of multiple people (for e.g., in the gym)
  We have tried to use images only with a single person.
- Lack of proper expert advice/definition on the correct posture posed a problem, as different people had varying (although slight) opinions.
  We have assumed that the narrator in the video is giving proper advice and used images which did not have contradictory advice as much as possible.
- Lack of consistent lighting conditions.

We collected a total of 601 images, out of which 375 images represented good posture and 226 images were bad posture.

3.2 Keypoint Estimation

In order to get the poses of people in images, we used human keypoint detection. We tried with different models such as Openpose, Microsoft Pose estimation, Keypoint RCNN for detecting keypoints in these images. After visually inspecting the output we decided to go ahead with the Keypoint RCNN model as results seemed better.

Fig. 7. Keypoints from OpenPose

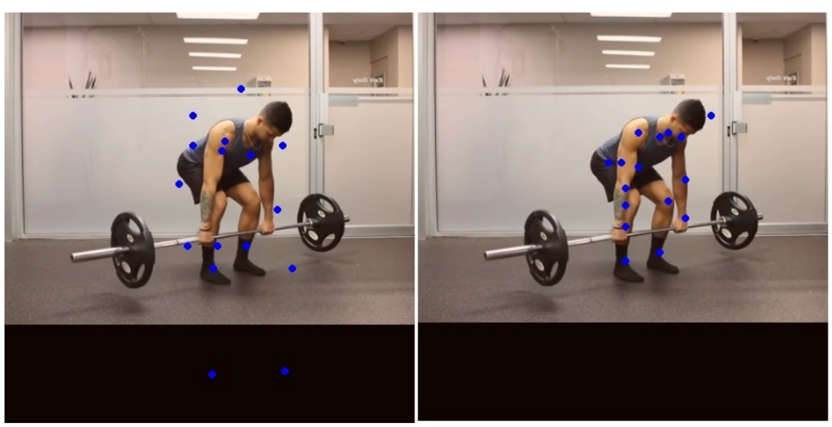

Fig. 8. Keypoints from Microsoft Pose estimation model

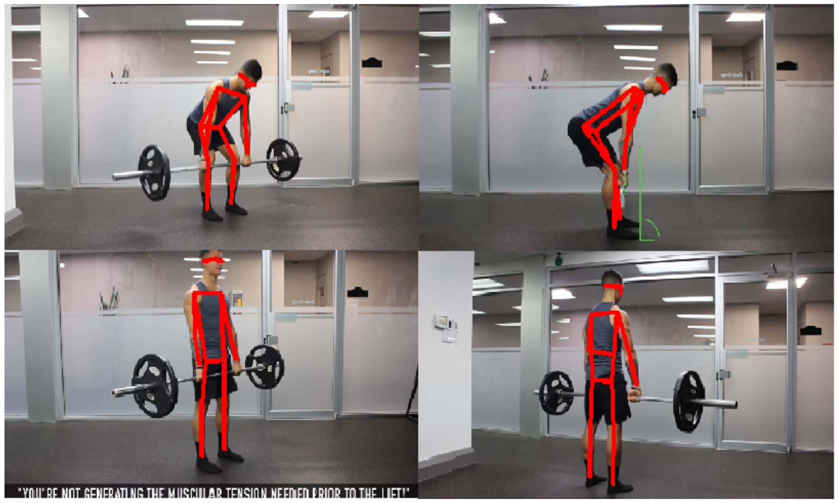

Fig. 9. Keypoints from Keypoint RCNN

After running RCNN model on our images, we proceeded to use the keypoints obtained, in two different ways for our model:

1. Use the keypoints generated on the fly as input for classifier(skeletal image)
2. Manually correct the keypoints (using a GUI) and use the corrected keypoint as input for classifier

3.2.1 GUI for correcting keypoints

We used the output of keypoint RCNN as baseline and implemented a python based GUI to correct the erroneous keypoints. Using the GUI, we saved the corrected skeletal image (with only the black background)

Fig. 10. Erroneous Keypoints

Fig. 11. Corrected Keypoints using GUI

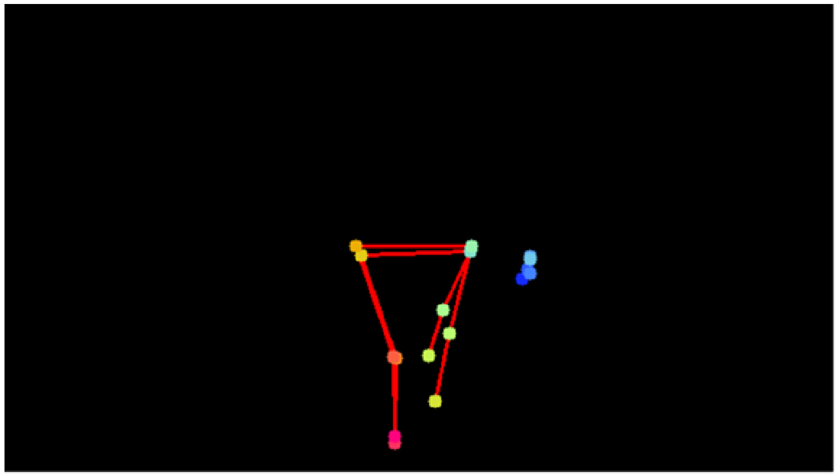

Fig. 12. Corrected skeletal image (which was saved)

Figure 10, represents the erroneous keypoints(error near ankle) got using Keypoint RCNN.
Figure 11, represents the manually corrected keypoint using the GUI
Figure 12, represents the skeletal image of the same on a black background, which was saved and used for future classifier.

3.3 Classification

We use the keypoints and the skeletal image obtained in the previous step in 2 different ways for classifying a posture as good or bad.

1. Resnet18 based CNN classifier: We use the skeletal structure obtained from keypoint estimation as the input image to a Resnet18 based CNN classifier.
2. DNN Classifier: The keypoints obtained are flattened into an array and fed to 4 layer deep neural network model.

3.3.1 Resnet18 based CNN classifier

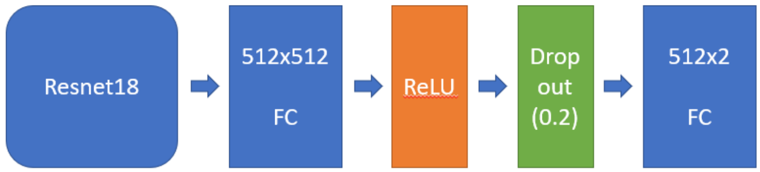

Fig. 13. Resnet18 based CNN classifier

We modified the Resnet18's fully connected layer by appending a sequence of 1 fully connected layer with ReLU activation, dropout and a a last layer with 2 outputs for 'Good' and 'Bad' posture.

3.3.2 DNN classifier

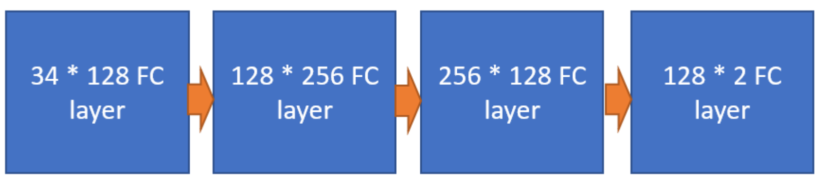

Fig. 14. DNN classifier

We implemented a simple deep neural network with the above mentioned parameters.

We tried generating the skeletal images on the fly and fed it as input to the both the classifiers sequentially. We also generated keypoints and manually corrected keypoints using the above-mentioned GUI and used these as inputs. We found the second approach to be slightly better.

4 Experimental Results

4.1 Resnet18 based CNN classifier

Learning Rate: 0.1
Epochs: 50 (Without early stopping)

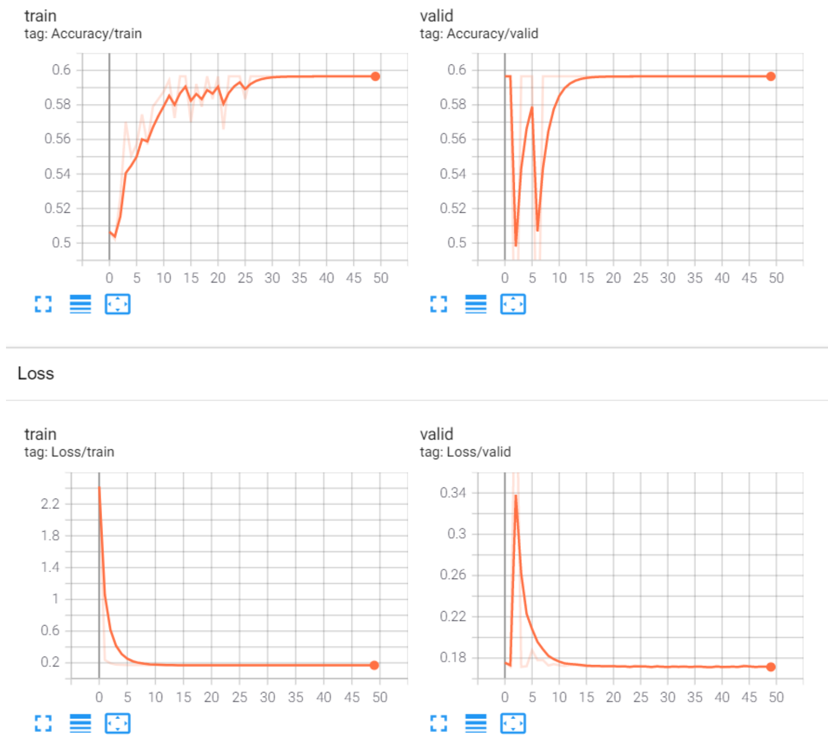

Fig. 15. Loss and Accuracy curves

From the curve we can see that both the training and test loss decreased initially but then saturated. Similarly for accuracy, the train accuracy increased up-to epoch 30 and the validation accuracy increased up-to epoch 15 and then have saturated.

Final accuracy
Training: 59.65%
Validation: 61.47%

4.2 DNN classifier

Learning Rate: 0.5
Epochs: 100 (in 2 parts)

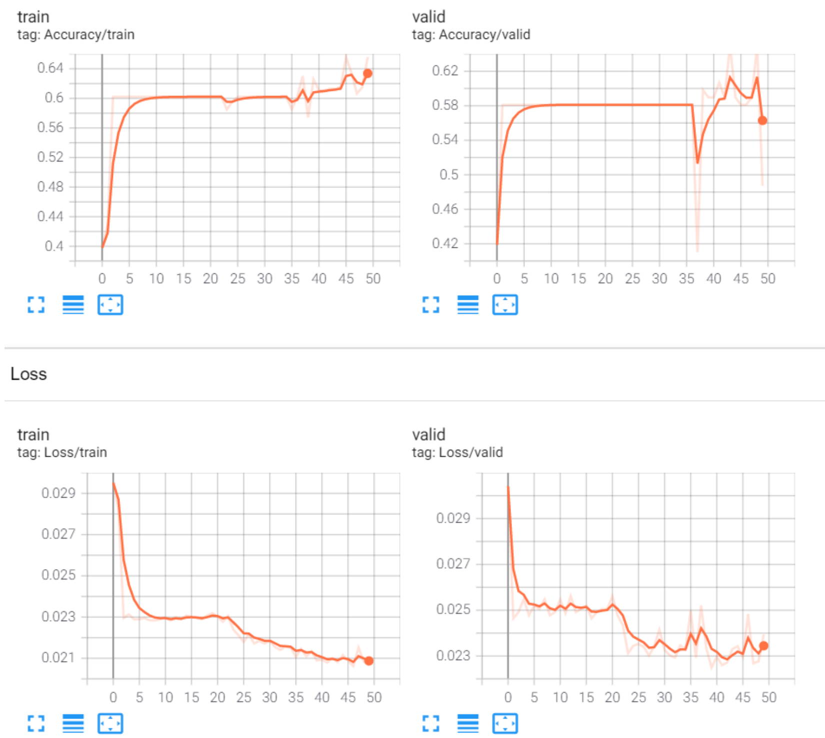

Fig. 16. Loss and Accuracy curves (first 50 epochs)

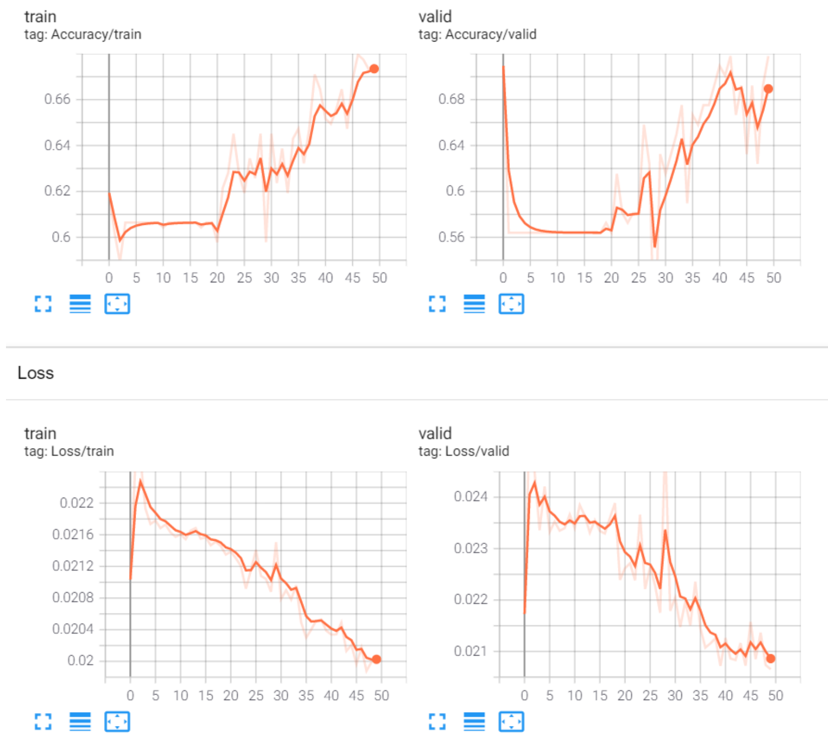

Fig. 17. Loss and Accuracy curves (50 to 100 epochs)

Training was done in 2 parts of 50 epochs each. Initially the model appears to saturate with respect to both accuracy and loss, but slowly starts to become better around epoch 25-35. But the learning remains inconsistent and erratic.
Final accuracy
Training: 67.53%
Validation: 71.79%

5 Conclusions and Future work

The model performed badly on both training and test datasets. The training and validation accuracy saturated in the case of CNN model and for DNN model, it is not consistent. Some of the reasons on why this could have happened are:

- Insufficient data: We collected only 601 images, which is not enough to train complex neural networks.
- Even for humans,the proper form is hard to distinguish: Different narrators of the video had slightly varying opinions about the correct form. This could have lead to potentially contradicting data.

- Wrong data format: We considered an image to determine the posture which might not have been the best format of data as it does not give the temporal relations of the keypoints.
- Shortcomings of Pose Estimation: Existing techniques do not capture the subtle curvature of human body, which is the key factor for deciding the proper posture.
- Using only the skeletal image with a black background is too sparse data for our model
- Using images with skeletal structure drawn on top of image adds too much data, and our model is not able to understand which part(of the image) is important.

A better approach would be to consider a data format which gives information about the relative motion of the keypoints with respect to time, like a video/gif and also collect more of such data. It might be better to encode the skeletal structure such that the neural network learns the points, their connections and what the point represents. Apart from pose estimation, it might be better to consider human segmentation as it better accents the human curvature.

References

1. Chen, K.: Sitting posture recognition based on openpose. IOP Conference Series: Materials Science and Engineering **677** (2019)

6 Assignment of Work

- Suhas Gopal
  - Worked on collecting data, GUI tool for annotation and generating skeletal images using Keypoint RCNN and manual annotation.
  - Implementing the DNN classifier.
- Suraj Sudhakar
  - Worked on collecting data.
  - Developing the GUI tool for annotation and manual annotation of data
- Harishanth Sivakumaran
  - Worked on collecting data, create a classification model.

7 Code Reference

1. Human Pose Estimation using Keypoint RCNN in PyTorch https://learnopencv.com/human-pose-estimation-using-keypoint-rcnn-in-pytorch/