A Mobile-Phone Pose Estimation for Gym-Exercise Form Correction

Matthew Turner[1], Kofi Appiah[1][a] and Sze Chai Kwok[2][b]

[1]*Department of Computer Science, University of York, Deramore Lane, York, YO10 5GH, U.K.*

[2]*Phylo-Cognition Laboratory, Division of Natural and Applied Sciences, Data Science Research Center, Duke Kunshan University, Duke Institute for Brain Sciences, Kunshan, 610101, Jiangsu, China*

{*mjt562, kofi.appiah*}*@york.ac.uk, szechai.kwok@duke.edu*

Keywords: Pose Estimation, Deep Learning, Mobile Application, Electronic Gym Instructor.

Abstract: People learn to perform exercises with good pose (or form) via research or instruction from an experienced individual such as a personal trainer, but 33.3% of injuries still occur due to incorrect form. It is known that the presence of a personal trainer causes a significant reduction in the rate that injuries occur. There are many possible reasons for this such as cost, scheduling limitations and desire to train alone. However, given that 91% of UK adults use a smartphone, a mobile APP could take on the role of a personal trainer. This paper presents a solution using machine learning and a novel proposed method of form anomaly detection to offer form corrections from live exercise video while only using the capabilities of a mobile device. Overall, the work in this paper is capable detecting incorrect exercise pose and offer valid corrections based on the detected anomalies. Experiments have been conducted on live video to judge the system performance in real-time.

1 INTRODUCTION

When exercising, your `'form'` refers to the way your body moves throughout the exercise. An example of a form requirement could be keeping your back straight during a dead-lift. Using correct form is critical for two main reasons, to avoid injury and to activate the intended muscles. According to a study done by *Summitt et al.* (Summitt et al., 2016) looking at shoulder injuries in Cross-fit, the most frequent causes of injury are the worsening/return of a previous injury (33.3%) and incorrect form (33.3%). This is a major issue, and a study by *Weisenthal et al.* (Weisenthal et al., 2014) shows that the presence of a trainer significantly decreases the rate at which injuries occur. During an exercise one would want activate (use) the intended muscles. Using dead-lift as an example, often when people do bend their spine during the lift, the back muscles are actually doing most of the work, even though the dead-lift should be a balance between the legs and the back. This and similar cases can result in muscle groups being under worked, and the individual not seeing the progress they expect. A study by *Kubo et al.* (Kubo et al., 2019) that compared muscle growth in a group performing the full squat (a squat with good form) and a group performing half squats (a squat with poor form), it was apparent that the difference in growth for certain muscle groups was as much as 4.5% on average. A simple remedy for this would be to have a trainer correcting your form.

2 RELATED WORK

The area of exercise analysis as a whole (ranging from exercise classification to exercise mistake detection) has started to see more attention as deep learning and other modern methods of pose estimation have gained some traction. In (Chen and Yang, 2020) the authors gauge correct form by using the angles between the limb vectors. For example, in a front raise they (Chen and Yang, 2020) measure the angle between the torso and the upper arm as one of the measures of success. The limb coordinates are acquired through deep learning with pose estimation (Chen and Yang, 2020). This method of using the angle between limbs does have some drawbacks. For example the method of mistake detection proposed in (Chen and Yang, 2020) is highly dependant on camera position, meaning that if the camera is not in the same plane as the angle being measured the angle will be inaccurate. Similar to (Chen and Yang, 2020), the work in (Yang et al., 2021) regularise the pose data using the average spine length. This is because users will naturally vary in

[a] https://orcid.org/0000-0002-9480-0679

[b] https://orcid.org/0000-0002-7439-1193

joint length, for instance a child will almost certainly have shorter limbs than an adult, this could impact the results greatly. One interesting point raised in (Yang et al., 2021) is the case of individuals with unusual body characteristics, such as a missing forearm, being something that the model might struggle with.

The work presented in (Rangari et al., 2022) is slightly different as it focuses on exercise recognition rather than form correction. The impressive results of 97.01% accuracy in (Rangari et al., 2022) demonstrates that exercise recognition is certainly possible and should be achievable with standard deep learning given enough data. In (Zhao et al., 2022) *Zhao et al.* apply 3D pose estimation, allowing for the user to be a lot more free with where they angle their camera. This is because the proposed model isn't trying to use angles, it's actually using a completely different approach consisting of two main steps and reported an average classification accuracy of 90.9%. The study in (Luna et al., 2021) was done by splitting the participants into two groups, one of which received supervision from a human physical therapist while the other received supervision from a mobile APP. The tests were done specifically on the squat for 3 sets of 10 in the order 10 unassisted, 10 assisted, 10 unassisted. The study concludes that there was no statistically significant difference between the group that received the human supervision as opposed to the group that received the AI supervision (Luna et al., 2021).

DeepPose (Toshev and Szegedy, 2013) is the first paper that applied deep learning to pose estimation. The most important point to note is involved with why deep learning is so suitable for pose estimation. In (Newell et al., 2016) propose the Stacked Hourglass Networks method, which is currently one of the premier standards for pose estimation. The proposed method in (Newell et al., 2016) gets it's name from the repeated cycles of pooling and up-sampling within the architecture, making an hourglass shape in terms of the layer dimensions. Crucially, the information from the pooling stage is combined into the up-sampling stage with the use of residual layers.

In (Zhang et al., 2018) *Zhang et al.* build upon the stacked hourglass network (Newell et al., 2016) by improving the speed at which the estimation happens with their proposed Fast Pose Distillation method. The work (Zhang et al., 2018) reports a mean accuracy of 91.1% on the MPII dataset across body parts, with 3 million parameters. The work in (Koskimäki and Siirtola, 2014) explores the idea of wearable sensors. Specifically the authors wanted to use data gathered by two accelerometers attached to the body in order to classify the exercise currently being performed by the wearer. In conclusion, the proposed method (Koskimäki and Siirtola, 2014) worked well on some exercises but struggled with others.

3 OUR APPROACH

In this work, the aim is to attempt to correct exercise form using machine learning with minimal resources. The method can be summarised as taking video of the subject, detecting the pose, then overlapping a theoretically perfect pose with the detected subject pose and giving feedback based on the differences. There are three main components to look at: the machine learning model, the method of detecting form anomalies and the binding of those two together in a manner that provides valid feedback.

3.1 Detecting Form Anomalies

To correct form, we naturally need a way of detecting when the form is incorrect. We proposes a new approach where a theoretically perfect pose is overlapped with the detected pose and feedback is generated based on the offset between set joints. For example, the offset between the theoretically perfect left shoulder location and the detected left shoulder location.

3.1.1 Detecting with Machine Learning

There are two approaches to detecting pose anomalies using machine learning. First a set of possible mistakes must be defined. Once this set has been defined, example images for each mistake need to be accumulated, as well as a set of images that show the exercise being performed correctly. The data is used to train a classification model that should be able to take input and detect which class the input falls into. This method (Rangari et al., 2022) (Zhao et al., 2022) also suffers from the common drawbacks of machine learning, such as the requirement for large amounts of training data and the complexity of introducing new classes (for new exercises) into an existing model without affecting the performance on the existing classes.

3.1.2 Detecting with Angles Between Joints

This method has been applied in at least two previous works (Chen and Yang, 2020),(Yang et al., 2021) but seems to have some significant limitations. Across the 8 classes (Chen and Yang, 2020) achieves on average a precision of 0.885, a recall of 0.860 and an F1 score of 0.858. This method does have some positives, out of all the methods that have been considered this one is the most flexible in terms of adding new exercises, as well as being able to give the clearest feedback. Unlike in (Rangari et al., 2022) and (Zhao et al., 2022), in this work, we hypothesise that incorrect form is identified by overlapping a theoretically perfect pose with the pose identified using machine learning. To understand the plan for pose overlapping, we must first consider how to set up the theoretically perfect skeleton. The skeleton will effectively be a collection of points in 3D space, each of these points will represent a joint in the human body. These joints will be the same as the ones the machine learning model detects. These points will be able to undergo a set of 3D transformations that step them through the perfect form for a specific exercise.

To overlap these points with the points detected by the machine learning model, the 3D points will need to be flattened into 2D points. In order to do this from the right angle the skeleton will need to be rotated in 3D such that the average point-to-point displacement between the detected points and the flattened skeleton points is minimised. In order to centre the theoretically perfect pose on the subject pose a central joint to match on will be chosen. The joints used will vary depending on the exercise. The feedback will be provided based on the point-to-point displacement as the video continues. To give an example, if the displacement of the shoulder joint from the theoretically perfect shoulder joint has been above a given threshold, a warning that the shoulder is in an incorrect position could be returned.

3.2 Pose Estimation Model

The Pose Estimation model chosen to be used in this project is the Fast Human Pose Estimation model proposed in (Zhang et al., 2018), this model builds upon the Stacked Hourglass architecture described in (Newell et al., 2016) by implementing the teacher-student approach to reduce the network size. This has the effect of making the model much faster while only slightly reducing the model accuracy. To implement the teacher-student approach you first train a larger, slower teacher model, then use the teacher model in the training process of a smaller, faster student model. The trained student model will be better than models trained from scratch at the same size.

In terms of up-sampling method, both (Newell et al., 2016) and (Zhang et al., 2018) use standard nearest neighbour up-sampling and achieve excellent results, and this work use the same concept. The loss function used in the teacher model is a Mean Squared Error (MSE) loss that gives the loss as the mean squared error between each value in the predicted heat-maps and each value in the ground truth heat-maps. It is worth noting that since the model prediction is a heat-map and the ground truth is a single point a fake ground truth heat map must also be generated. According to (Zhang et al., 2018) the best method here is using a 2D Gaussian with $\sigma = 1 pixel$

The student model as proposed in (Zhang et al., 2018) is just a smaller (and therefore faster) version of the model proposed in (Newell et al., 2016). The teaching process happens in the loss function, in (Zhang et al., 2018) they define their loss function as:

$$L_{fpd} = \alpha L_{pd} + (1-\alpha) L_{mse} \tag{1}$$

where L_{mse} is the normal loss as described in (Newell et al., 2016) but L_{pd} is the pose distillation function defined in (Zhang et al., 2018) as:

$$L_{pd} = \frac{1}{K} \sum_{k=1}^{K} \|m_k^8 - m_k^t\|_2^2 \tag{2}$$

where:

- k is a single joint in the set of joints with length K
- m_8^k is the confidence map for the k-th joint as predicted by the teacher
- m_k^t is the confidence map for the k-th joint as predicted by the student

For training data, the model will be trained using the MPII Human Pose dataset (Andriluka et al., 2014). This is a commonly used dataset for human pose estimation also used in (Newell et al., 2016), so it is known that it produces good results in combination with the chosen method. An idea brought up in (Yang et al., 2021) also suggests inputting data at different rotations to prepare for situations where the user is in an unusual orientation (such as during a handstand), as there is potential for this to come up in many exercises, rotation augmentation will be applied to the training data. Examples from the training data where multiple people are present will be filtered out. This is because in this work, we only expect a single subject, as that's all that will be needed for the use case. However, examples where certain joints are fully occluded will remain in the data, as this is a situation that the model will need to be prepared for.

3.2.1 Evaluation

There are multiple options for evaluating the performance of human pose estimation models as clearly outlined in (Barla,). Having considered the options, 'percentage of correct points' (PCK) has been chosen for this work as it is the most commonly used method that provides metrics for each joint individually. There are two types of PCK, PCKh@0.5 and

PCK@0.2, in this work PCKh@0.5 will be used, as it is the most widely used. PCKh@0.5 is a metric that defines a joint as 'found' if the distance between the detected joint and the true joint is less than $0.5 \times head\_bone\_link\_length$. The PCKh@0.5 score across a set of data is defined for each joint as $joint\_found\_count/total\_joint\_count$.

3.3 Theoretical Perfect Pose

In order to implement a theoretically perfect pose a skeleton that could be transformed in 3D space was required. This skeleton could be transformed such that it steps through the exercise and then flattened into 2D for overlapping. To create the skeleton, a Joint class is needed to represent each joint in the skeleton where instances of this class correspond with joints in the training data. Every instance of the Joint class can have child instances, where transformations applied to the parent instance will also apply to the child instances. This is so the body moves logically, for instance if the elbow rotates the hand must also rotate. The main components of each instance of the Joint class are therefore the position in 3D space (x, y and z coordinates) as well as a list of children (other instances of the Joint class).

When the entire skeleton is rotated, unit vectors axes are rotated along with it. Thus if a skeleton instance is rotated 90° around the y axis, the relative axes are also rotated. So if a new rotation is applied to the arm of the skeleton, instead of rotating around the 'true' x axis, the rotation occurs around the relative x axis. This also necessitated a new function in the Joint class that can rotate around any unit vector in 3D space, rather than around the true axis.

3.4 Pose Estimation Model

A DataHandler class was defined to load the MPII Human Pose (Andriluka et al., 2014) annotations and store them. Some data filtering also occurred at this step, namely the removal of images with multiple people. This was with the intention of speeding up training by making the problem simpler. Furthermore, since the user can move the camera, it is expected that there should only be one person centered in the foreground. The DataHandler class is also where the image loading and preprocessing occurs, this uses a python library called imgaug (Jung, 2020) in order to augment the input data while keeping track of the key-points (in this case joints) on the image. The preprocessing for training consists of:

- Cropping around the person
- Padding the image to square (padding with black pixels)
- Resizing to model input dimensions
- Random rotation between -30° and 30°

Input batches consist of the input images as well as the 16 joint heat-maps. Therefore, the PoseEstimationDataset class also owns an instance of the HeatmapGenerator class. This class generates the joint heat-maps according to (Zhang et al., 2018).

In order to test the model a ModelEvaluator class was defined, this class could be expanded to evaluate the model in many ways but the one implemented in this work is testing to get the PCKh@0.5 (Yang and Ramanan, 2013) score for each joint. To get the PCKh@0.5 score for each joint the class is given a set of images featuring a single person, the set of true joint locations and a trained model. The output from the model on the images is evaluated against the true joint locations to get the PCKh@0.5 score for each joint. To overlap the skeleton onto the detected joints, they are first matched at a single joint. This joint depends on the exercise, for example the pelvis joint is used when overlapping a squat. Once this single joint is matched the skeleton undergoes a series of transformations with the aim of minimising the average distance between skeleton joints and detected joints. Once the perfect pose is overlapped with the detected pose, feedback can be generated.

4 EVALUATION - SUCCESS CRITERIA

To accurately identify mistakes in exercise form from an input image, we have the following criteria:

- Achieve an average PCKh@0.5 score of > 0.9 on visible joints in MPII Human Pose (Andriluka et al., 2014) test data.
- Achieve an average PCKh@0.5 score of > 0.9 on visible joints in example input data.
- Skeleton overlaps onto detected pose as expected.
- Skeleton-overlap reports joints in incorrect position correctly with a false correct rate of 0.0.
- Skeleton-overlap report joints with a false incorrect rate of < 0.1.

To generate valid corrections based on detected mistakes, the following are the required criteria:

- When a mistake is detected the reported corrections contain a fix to the mistake.
- None of the reported potential corrections should cause further mistakes.

- None of the reported corrections should be irrelevant to the current exercise.

The following criteria is used to identify mistakes in an exercise and generate valid corrections in real-time using live video stream:

- The required memory to process the input video stream should not exceed 4GB, to guarantee a successful implementation on a mobile device.
- Ability to accept live video stream and output correction instantaneously as mistakes occur.

4.1 Performance on Testing Data

To start the evaluation both the large model and the small model were ran on a set of testing data from MPII Human Pose (Andriluka et al., 2014) to get the PCKh@0.5 scores for each joint. There are two output rows in tables 1 and 2; the ”Visible Only” row where joints that are not visible in the image are considered found by default and the ”Including Non-Visible” row where joints that are not visible are expected to be output as a blank heat-map.

We can see from tables 1 and 2 that while the large model achieves an average visible-only PCKh@0.5 score of 0.696 the smaller model actually achieves a slightly better average of 0.697. This indicates that for this training data, the additional model size is not necessarily beneficial. Both models perform poorly when detecting non-visible joints compared to only visible joints, they do not understand that the joint cannot be found and instead choose the most likely location.

4.2 Testing on Example Input Video

The small model was adopted for these video based tests. Three exercises were selected at varying levels of complexity to gauge performance across a range of scenarios, these exercises were:

- Bicep half-curls: a variant of the bicep curl where the forearm pauses when parallel to the ground.
- Lateral raise: an exercise where a dumbbell is held in each hand and lifted outwards through the scapular plane to just below parallel.
- Barbell squat: where a barbell is placed across the shoulders while the individual squats down.

Video for each of these exercises was acquired in a real gym to allow for testing with realistic background imagery. To visualise the results on video the results on 5 key-frames throughout the exercise will be shown. The detected joint locations will be marked with coloured circles. The green circle indicates the detected pelvis joint location while the blue circles represent all other joints. These frames have been hand-annotated to get a PCKh@0.5 score.

4.2.1 Bicep Half-Curl Video

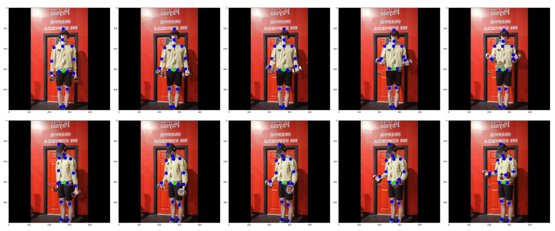

Figure 1: Predicted joint locations on two videos of half-curls.

From figure 1 and table 3 we can see that model performance on the half-curl data is very good, achieving an average PCKh@0.5 score of 0.981 on the 10 test frames. The difference between this average and the average on the MPII Human Pose (Andriluka et al., 2014) test data is likely due to the location of the individual being in the centre of the image. The MPII (Andriluka et al., 2014) data does not necessarily have the individual in the centre of the image, but the training data was preprocessed to make this the case, thus the model performs significantly better on centred individuals.

4.2.2 Lateral Raise Video

The results for lateral raise as shown in table 4 are good, achieving an average PCKh@0.5 score of 0.963 on visible-only joints and a slightly lower average PCKh@0.5 score of 0.838 when non-visible joints are included. Looking at figure 2 we can see that the ankles of the individual performing the exercise are not visible in any of the frames, this is likely the cause of the difference. Another point to note is that the model seems to struggle slightly with the wrist detections in this instance. This could be due to the presence of a dumbbell in the hands of the individual that is slightly obscuring the wrist. Another potential reason could be that the wrist is not very central to the image, which would typically be the case in the training data.

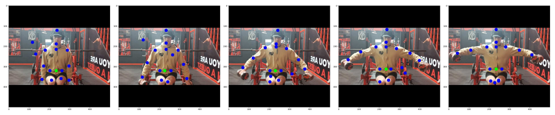

Figure 2: Predicted joint locations on a video of a lateral raise.

4.2.3 Squat Video

From table 3 alone it seems that model's performance on the squat data is acceptable, the average

Table 1: Large (8 hourglass) model PCKh@0.5 scores for each joint.

	r_ankle	r_knee	r_hip	l_hip	l_knee	l_ankle	pelvis	thorax	neck	head	r_wrist	r_elbow	r_shoulder	l_shoulder	l_elbow	l_wrist	average
Visible Only	0.753	0.679	0.669	0.67	0.68	0.75	0.783	0.827	0.627	0.564	0.678	0.667	0.716	0.721	0.673	0.679	0.696
Including Non-Visible	0.213	0.273	0.336	0.34	0.284	0.219	0.27	0.383	0.625	0.562	0.448	0.46	0.481	0.472	0.447	0.447	0.391

Table 2: Small (2 hourglass) model PCKh@0.5 scores for each joint.

	r_ankle	r_knee	r_hip	l_hip	l_knee	l_ankle	pelvis	thorax	neck	head	r_wrist	r_elbow	r_shoulder	l_shoulder	l_elbow	l_wrist	average
Visible Only	0.752	0.681	0.66	0.653	0.67	0.751	0.781	0.841	0.658	0.603	0.677	0.653	0.721	0.725	0.666	0.667	0.697
Including Non-Visible	0.213	0.276	0.327	0.323	0.274	0.223	0.268	0.397	0.655	0.6	0.447	0.445	0.486	0.477	0.44	0.436	0.393

Table 3: PCKh@0.5 scores for the bicep half-curl predictions.

r_ankle	r_knee	r_hip	l_hip	l_knee	l_ankle	pelvis	thorax	neck	head	r_wrist	r_elbow	r_shoulder	l_shoulder	l_elbow	l_wrist	average
0.9	1.0	0.9	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.9	0.981

Table 4: PCKh@0.5 scores for the lateral raise predictions.

	r_ankle	r_knee	r_hip	l_hip	l_knee	l_ankle	pelvis	thorax	neck	head	r_wrist	r_elbow	r_shoulder	l_shoulder	l_elbow	l_wrist	average
Visible Only	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.8	1.0	1.0	1.0	1.0	0.6	0.963
Including Non-Visible	0.0	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0	0.8	1.0	1.0	1.0	1.0	0.6	0.838

PCKh@0.5 scores of 0.838 on the visible-only data and 0.712 on the including non-visible data are good, but looking at figure 3 we can see that in the two right-most images joints have been detected hovering in the air above the individual. The reason for the increase in issues as we move through the squat is likely due both to the individual leaving the centre of the image (as they lower through the squat) and the unusual body shape of the squat which isn't likely to be featured in the training data.

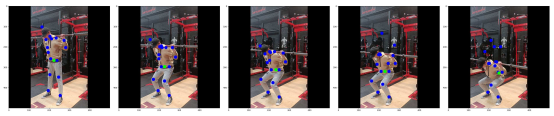

Figure 3: Predicted joint locations on a video of a squat.

4.2.4 Overall Performance on Input Video

Across the three exercises that were tested sections 4.2.1, 4.2.2 and 4.2.3, the model achieved an average PCKh@0.5 score of 0.927 on visible-only data and 0.844 when non-visible joints were included. This indicates that the model performs at a good level on visible joints, but will struggle to infer a reasonable joint location when the joint is not visible. Considering how the model fits into the application, this indicates that the overall application will likely perform well when the joints relevant to the exercise are visible, but will struggle if relevant joints are hidden.

4.3 Test for Skeleton Overlap

We investigate if the implementation is capable of overlapping the skeleton onto the poses detected and then visually evaluated as 'expected' or 'incorrect'.

To clarify, figure 4 shows on the left an input image with the detected joint locations highlighted.

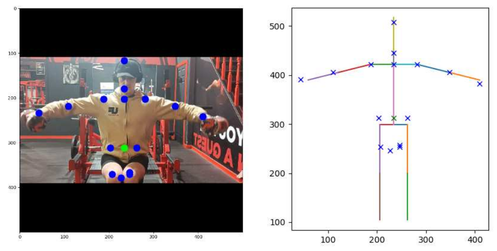

Figure 4: Example of a set of pose detections and the corresponding skeleton overlap.

Those joint positions are then moved onto the plot on the right where the skeleton is overlapped onto them using the overlapping algorithm explained in the implementation. In this example because the exercise is a lateral raise, the only relevant joints are the shoulders, elbows and hands. By comparing the input image and the skeleton overlap we can see that in this case the overlap has worked, so this would be labelled visually as 'expected'.

4.3.1 Skeleton Overlap on Bicep Half-Curl

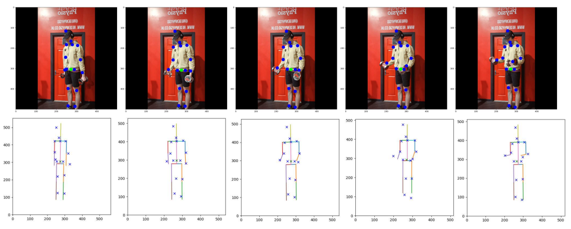

Figure 5: Comparison showing the detected joint locations on the input half-curl frames and the skeleton overlapped onto the detected pose.

Looking at figure 5 we can see that for each image, the skeleton is overlapped correctly. All the images tested in this scenario resulted in an 'expected' output.

Table 5: PCKh@0.5 scores for the squat predictions.

	r_ankle	r_knee	r_hip	l_hip	l_knee	l_ankle	pelvis	thorax	neck	head	r_wrist	r_elbow	r_shoulder	l_shoulder	l_elbow	l_wrist	average
Visible Only	1.0	0.8	0.8	0.4	0.4	0.8	0.8	1.0	1.0	1.0	1.0	0.6	0.8	1.0	1.0	1.0	0.838
Including Non-Visible	1.0	0.8	0.8	0.4	0.4	0.8	0.8	1.0	1.0	1.0	1.0	0.6	0.8	0.4	0.4	0.2	0.712

4.3.2 Skeleton Overlap on Lateral Raise

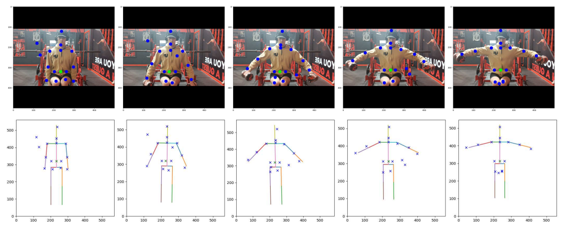
Figure 6: Comparison showing the detected joint locations on the input lateral raise frames and the skeleton overlapped onto the detected pose.

Looking at figure 6 we can see that the skeleton overlaps are as expected for every image, the skeleton lateral raise progress successfully mirrors the progress of the individual performing the exercise. The legs do not line up in this case because as this is a lateral raise, the leg joints have no relevance to correct form.

4.3.3 Skeleton Overlap on Squat

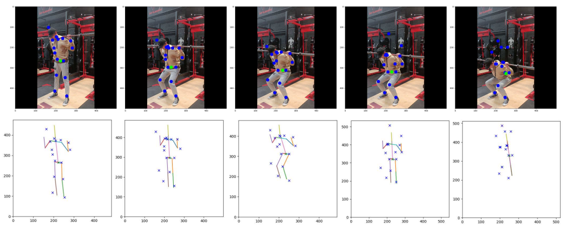
Figure 7: Comparison showing the detected joint locations on the input squat frames and the skeleton overlapped onto the detected pose.

Looking at figure 7 we can see that some of the skeletons have not been overlapped correctly. Namely (moving from left to right) the second, fourth and fifth images. All of these overlaps failed due to incorrectly detected joints. The second is hard to spot but it failed because the right wrist was not correctly detected, the fourth and fifth are easier to see as we can see detected joints floating above the individual. To clarify what has happened in the skeleton overlap for the fifth image, the detected optimal rotation for the individual has rotated them so that we are viewing them from the side. As the skeleton is flat, this side view has appeared as a line when flattened from 3D to 2D.

4.4 Testing Joint Incorrect/Correct Reporting

In order to evaluate whether joints in incorrect positions are reported as incorrect and joints in correct positions are reported as correct, a small test set was accumulated consisting of both images of exercises being performed correctly and of exercises being performed incorrectly. The images were ran through the implementation and either detected as having an incorrect joint or not, and the performance evaluated.

Table 6: Confusion matrix showing exercise report results.

	Ground Truth		
	Correct	Incorrect	Total
Correct	14	2	16
Incorrect	6	9	15
Total	20	11	31

From the confusion matrix (table 6) we can calculate the false correct rate (FCR) and the false incorrect rate (FIR).

$$FCR = \frac{FalseCorrect}{FalseCorrect + TrueIncorrect} = 0.182 \quad (3)$$

$$FIR = \frac{FalseIncorrect}{FalseIncorrect + TrueCorrect} = 0.300 \quad (4)$$

4.5 Test on Memory Usage

In order to test if the implementation will fit on a memory constraint mobile device, five minutes of video data was ran through the full system's pipeline. During this the RAM allocated to the Python process and the GPU memory allocated to PyTorch (Paszke and et. al, 2019) was measured at each frame. Figure 8 shows the graph of memory usage over the frames.

On figure 8 we can see that the 4GB limit specified as the threshold for a mobile device was never exceeded. Throughout the entire process the memory usage remains comfortably below that mark.

5 CONCLUSION

The overall goal of this work was to develop an alternative to the traditional personal trainer using machine learning as well as a novel new method of detecting exercise form mistakes. After evaluating the

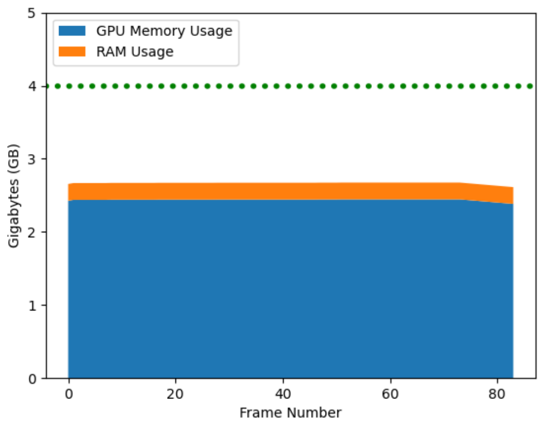

Figure 8: Stacked plot of the RAM usage and GPU memory usage of the application while running on video input

performance of the implemented solution, we have demonstrated that detecting exercise form on a mobile device can be achieved. The first aim of this work was to accurately identify mistakes in exercise form from video input. To recap, the proposed method was to overlap a 'perfect pose' skeleton onto the detected pose of the individual performing the exercise. From the evaluation we can see that when even a single joint was detected incorrectly, because of the chosen method to overlap the 'perfect pose', the overlap would be misaligned and report multiple incorrect joints. Even if all the joints are detected correctly the alignment could still go wrong due to issues with overlapping a 3D object onto 2D points, mainly not being able to try every combination of rotations due to processing limitations.

The second aim of this work was to generate valid corrections based on detected mistakes. The third aim of this work was to use live video as input to perform the form correction on a mobile device. The issue with running on live data was that the proposed solution took too long to run a single frame due in part to the machine learning model delay but mostly due to the delay caused by the overlapping algorithm. From this summary, we can see that the main issues with this work is centred on the overlapping algorithm. This process had issues both in terms of accuracy and duration. Potential avenues for future research could include alternatives to this algorithm while still retaining the concept of overlapping the 'perfect pose'.

REFERENCES

Andriluka, M., Pishchulin, L., Gehler, P., and Schiele, B. (2014). 2d human pose estimation: New benchmark and state of the art analysis. pages 3686–3693.

Barla, N. A comprehensive guide to human pose estimation.

Chen, S. and Yang, R. R. (2020). Pose trainer: Correcting exercise posture using pose estimation.

Jung, A. (2020). imgaug. *GitHub repository*.

Koskimäki, H. and Siirtola, P. (2014). Recognizing gym exercises using acceleration data from wearable sensors. *2014 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 321–328.

Kubo, K., Ikebukuro, T., and Yata, H. (2019). Effects of squat training with different depths on lower limb muscle volumes. *European Journal of Applied Physiology*, 119:1933–1942.

Luna, A., Casertano, L., Timmerberg, J., O'Neil, M., Machowsky, J., Leu, C.-S., Lin, J., Fang, Z., Douglas, W., and Agrawal, S. (2021). Artificial intelligence application versus physical therapist for squat evaluation: a randomized controlled trial. *Scientific Reports*, 11.

Newell, A., Yang, K., and Deng, J. (2016). Stacked hourglass networks for human pose estimation. *CoRR*, abs/1603.06937.

Paszke, A. and et. al (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Rangari, T., Kumar, S., Roy, P. P., Dogra, D. P., and Kim, B.-G. (2022). Video based exercise recognition and correct pose detection. *Multimedia Tools and Applications*, 81:30267–30282.

Summitt, R. J., Cotton, R. A., Kays, A. C., and Slaven, E. J. (2016). Shoulder injuries in individuals who participate in crossfit training. *Sports Health*, 8:541–546.

Toshev, A. and Szegedy, C. (2013). Deeppose: Human pose estimation via deep neural networks. *CoRR*, abs/1312.4659.

Weisenthal, B. M., Beck, C. A., Maloney, M. D., DeHaven, K. E., and Giordano, B. D. (2014). Injury rate and patterns among crossfit athletes. *Orthopaedic Journal of Sports Medicine*, 2.

Yang, L., Li, Y., Zeng, D., and Wang, D. (2021). Human exercise posture analysis based on pose estimation. *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference*, pages 1715–1719.

Yang, Y. and Ramanan, D. (2013). Articulated human detection with flexible mixtures of parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2878 – 2890.

Zhang, F., Zhu, X., and Ye, M. (2018). Fast human pose estimation. *CoRR*, abs/1811.05419.

Zhao, Z., Kiciroglu, S., Vinzant, H., Cheng, Y., Katircioglu, I., Salzmann, M., and Fua, P. (2022). 3d pose based feedback for physical exercises.