HOME FITNESS COUNTING AND ESTIMATION SYSTEM

Ding Yuxing, Sun Chengyuan, Wang Futong, Xu Mingjie

Institute of Systems Science, National University of Singapore, Singapore 119615

ABSTRACT

We present a Home Fitness Counting and Estimation System, a system with three models that can help people better monitor and adjust their daily exercise in current stay-at-home policy by getting the fitness video as the input and give classification, counts and the score of each repeat as the result. Our contributions include a YOLO human detection model to crop the particular area, a model for counting the repeating motions based on RepNet and a KNN based motion classification and estimation model using the human body key points extracted through mediapipe framework.

***Index Terms*—** Human detection, repeat counting, pose estimation, fitness motion evaluation.

1. INTRODUCTION

In most of the countries at present, the shadows of COVID-19 is yet not decayed. Humans beings are trying best to fight against the virus and get back the normal life but this would be a gradual process to take and during the time we still have to stay at home according to the policies to reduce the chance of infection and ensure safety. Fitness in this time period is important as it can not only keep us in a healthy physical body status, but also in mental release us from the depression created by long term stay in concealed and lonely environment. We notice that it has been a trend that people start to follow videos or some fitness apps like Keep to achieve daily training task which is usually broken down into many small parts of repeating motions for particular times like squat, push up and so on. The issue is, unlike in the gym where the coach can give some suggestions, during the training, people just follow those trials but actually they won't have a chance to know how did they perform in the past sessions. As the exercise pushes further and people get more tired, even counting would be a hard task to do. We think it will be an interesting and meaningful topic to build up a system that could help to evaluate the quality of fitness training, as this would help people to promote efficiency during the physical training.

In this system, we take a video of one repeating motion as an input and output another video with counting, motion classification and score of each repeat labeled in frames.

Firstly the video will be analyzed by a human detection model to crop the area of human body only, and then restore it into a video of a certain resolution according to the cropping result of the video. If there are multiple people doing fitness together, the result would be each video for each person. Then the video will be sent to the RepNet model to count the quantity of repeating and record the timestamps of each motion. At last, according to the timestamp, each motion will be sent to the pose estimation and evaluation part and receive label and score. The final output video will be the original one with labels and scores tagged on each frame.

2. LITERATURE REVIEW

2.1. Human Detection

YOLO:the reference can be SSD method and faster RCNN.

Table 1. SSD and YOLOV3 comparison.

	SSD	YOLOV3
Loss	Softmax loss	Logistic loss
Feature extractor	VGG19	Darknet-53
Bounding Box Prediction	direct offset with default box	offset with grid cell by sigmoid activation
Anchor box	Different scale and aspect ratio	K-means from coco and VOC
Small objects	Semantic value for bottom layer is not high. Worse for small objects.	Higher resolution layers have higher semantic values. Better for small objects.
Big objects	Better. Feature map rangers from 38 * 38 to 3 * 3 ,1 * 1	Worse. 13 * 13 feature map is the most coarse-grained.
Data Augmentation	different sample IOU crop on original image	randomly put the scaled original image (from 0.25 to 2) on the gray canvas
Input	resize original image to fixed size	Random multi-scale input
FPN	no	with FPN

In the loss of SSD, the classifiers of different categories are softmax, and the category of the final detection target can only be one category. In yolo-v3, for example, for the 80-category coco data set, 80 logistic classifiers are used to determine the category. As long as the output is greater than the set threshold, they are all categories of objects. Objects can belong to multiple categories at the same time, such as one The objects are both person and woman.Backbone network. The basic network of the original ssd is VGG19, and mobile-net, resnet, etc. can also be used. The basic network of yolo-v3 is darknet-53 (because it has 53 convolutional layers) designed by the author himself, borrowing from the shortcut layer of resnet,according to the author, a similar effect is achieved with fewer parameters and less calculations.

Fast RCNN: fast RCNN uses a network to implement all parts except the region proposal extraction. Unlike RCNN, 1) his classification and coordinate regression loss are used to update the network parameters through backpropagation; 2) it is in When extracting features, each region proposal is not included in the extraction, but after extracting the features of the entire image, the feature is extracted by coordinate mapping. This has two advantages, a) fast, because only one image is taken Primary network; b) The feature's feature is affected by the receptive field, and it can merge the features of the adjacent background, so that it can "see" farther.

Pros and cons Advantages: fast, simple pipeline; low background false detection rate; strong versatility, YOLO is also suitable for object detection of artistic varieties. Its detection rate of unnatural image objects is much higher than DPM and R-CNN series detection algorithms. However, compared with the R-CNN series of object detection methods, YOLO has the following shortcomings: the accuracy of identifying the object position is poor; the recall is low, and the constraint method of predicting a fixed number of bboxes in each grid reduces the number of candidate frames. YOLO[1] is better for our Project.

2.2. Repition Counting

For repnet, the basic structure is provided by the paper[2]. The whole structure of the model is a 2D CNN feature extractor , followed by a 3D CNN, then followed by a self similarity matrix and final it's a transformer with 2 MLP classifier.It is very straightforward, CNN layer is doing feature embedding, and transformer layer is for learning from the time sequence.Also with the help of similarity , it obviously can find the similarity frames in a time sequence so call the repetition. In our project, we borrow their idea on this repetition predictor, but ours is a bit different. Firstly, we use MobileNet as backbone for 2D feature embedding, because we want our model is as sampler as possible as our scenario is always in door activities. Secondly, we use two transformer layers respectively for the two classifier at the end, because the significant loss reduction happens earlier than a shared transformer layer when training.

To classify and evaluate the motion, the first thing we need to do is applying pose estimation to get the key points of human body in a 2D image. In this area, [3]OpenPose has been widely used as its average performance among different datasets is pleasant. But last year in 2020, google released their new framework mediapipe, which has a pose module based on the [4]BlazePose, and it is obviously faster than OpenPose and beats OpenPose in some datasets like Yoga and Fitness Pose. We choose BlazePose as the pose estimation model since it is easy to apply and its strength on the Fitness poses dataset can meet our application occasions. The following table shows the performance of the above models on AR and Yoga datasets:

Table 2. The performance comparison with OpenPose.

Model	OpenPose	BlazePose
FPS	0.4	10
AR Dataset, PCK@0.2	87.8	84.1
Yoga Dataset, PCK@0.2	83.4	84.5

2.3. Pose Estimation and Classification

BlazePose detects 33 human body key points in total, 16 more compared with OpenPose using coco, which enables the information inference of ratio and directions of arms and legs in a simulated 3D space. During the training, the author applied blocking simulation augmentation to reduce the affect of key points lost in the training set. The weakness of BlazePose is that it is not able to tracking multiple human bodies as the training sets are all single-person scenario. However, with the help of human detection model, we can ensure that the image sent to the model has only one person.

For the action classification part, the usual method come into people's mind is deep learning method taking the frames as input and inferring the content from the motions. One of those models we referred is [5]RC3D, R-C3D: Region Convolutional 3D Network for Temporal Activity Detection, a time sequence based temporal action detection model. The 3D ConvNet takes raw video frames as input and computes convolutional features. These are input to the Proposal Subnet that proposes candidate activities of variable length along with confidence scores. The Classification Subnet filters the proposals, pools fixed size features and then predicts activity labels along with refined segment boundaries. Generally the feature would be based on the whole image not the human alone and if we want to focus on human body only, the segmentation for the training set would be a tedious and huge work to do. RC3D performs relatively good among those similar action recognition models, however in the paper, the mAP@0.5 is less than 0.5 in all the action cases, which is not that confidential for the use case.

There are also other methods like Action recognition based on 2D skeletons extracted from RGB videos [6], extract key points from OpenPose and then proceed a CNN model to classify the key points sets of frames. The problem of those deep learning method is, they always requires a lot of data for training and a very powerful GPU to train and inference. Also, inference an action by a flow or sequence is not reasonable, as in the fitness pose what we require is that the key actions should be correct. Thus we choose KNN, a traditional machine learning methods to classify those key frames. It could be fed with less data compared with deep learning method. Although it would be weaker as we increase the number of classifications, but for our case this is acceptable, as the daily exercise won't be that complex.

3. DATASET

3.1. Dataset for YOLO

We finally choose the official pre-train model.But we still did a training on yolov3, although the performence was not good For trained by ourselves part: We used a training set is" VOCdevkit"(852M) as show in Figure 1.Which Include Annotations, ImageSets, JPEGImages, SegmentationClass and Segmentation Object. Annatation record the image basic information and the object name as label. ImageSets is to split dataset to test/train/train valadation/test valadation set by imageid. JPEGImages contain 9963 images with different kinds of objects. SegmentationClass and Segmentation Object is processed image.

To test our YOLOV3 on our project, we will record our developer selves excise video to evaluate the object detection.The test result can be found in 5.Experimental Results.

3.2. Dataset for Period estimator

Currently, there are few public data sets of repetition videos. One is QUVA [7] which only contains 100 videos, the other one is Kinetics[8] data sets which is around 90 times times bigger than QUVA. But unfortunately, this data sets is only provide URL for videos on YouTube and many of them are invalid. Also it's very difficult to label all the frames while only the count number is provided. In this case, we take the idea of Synthetic Repetition Videos[**?**] to prepare videos for this part. There are 2 main advantage of the synthesis video. On one hand, synthesis video are all created from real videos so it's very close to real repeated videos when compared to using synthetic patterns. On the other hand, the diversity of data seen by the model can huge and the class labels can be as balance as possible.

There should be general 3 steps to generate a synthesis repeated video.Firstly, sampling a random video from a dataset of videos. In our project, we use around 1700 video download from YouTube. Secondly, sample a clip of random length

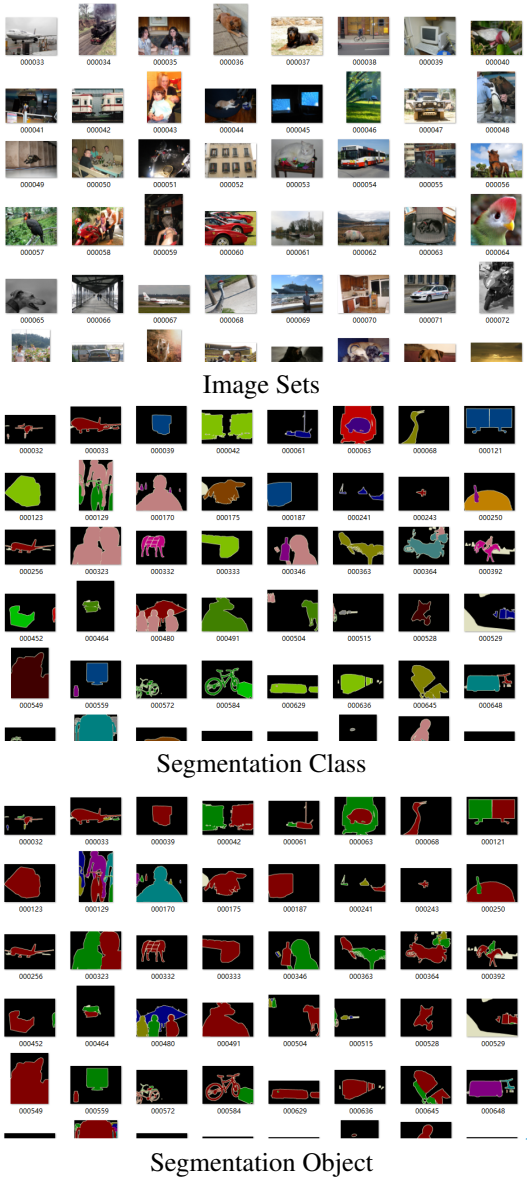


Fig. 1. YOLO ImageSets.

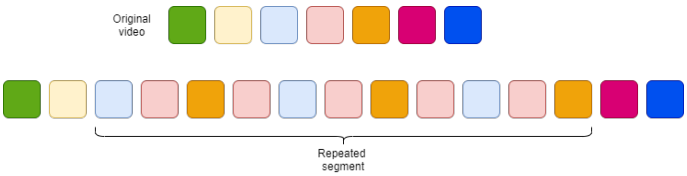


Fig. 2. synthetic video.

Fig. 3. Random rotation with blender.

frames from the video and then repeated K times as repetitions. And it's very important to concatenate the reversed clip Figure 2 before repeating , because in real world, motion is always done in reverse. Finally, put a non-repeating segments before or after repeating part, the length is random and also can be zero. Actually to simulate the real situation, 80% of the videos only have non-repeating segments, because video always need to be split before put into the model.

Before we use those videos to train the model, we need to do data augmentation including rotation, spatial shift, spatial zoom. But augmentation are not totally random. Firstly we generate a seed for a video, then apply time sequence with this seed to generate array as the transformation parameter for every frame.Beside we also add continuous and irregular moved blender mask in every frame.Sample frames is shown in Figure 3

3.3. Dataset for Pose

Manually collected data (3 motions, each 2 key frames, each key frames hundreds of manually tagged image, with data augmentation)

The dataset we used for pose classification part is manually collected and created from video websites as well as fitness tutorial app. We first determined that the fitness poses we could like to process are: squat (key pose: squat up, squat down), jump jack(jump open, jump close) and push up(push up, push down). Then for each of the motions pick one video, manually extract the key poses in different angles(if applicable) as the initial data. Below the Figure 4 shows the initial sample images of one of the key poses, jump jack open.

Then we will run the training based on those few images to get a model to help us to collect images. We set up the threshold a small number to capture more data for further selection.Then, search on youtube or other video website and download the target videos, run a python script to analyse them by frame and sent the images that the scores are above threshold to the categorised folders. And at this stage, more images are collected. Before move to the next stage, we need to mannually clear the data first as a lot of repeating or bad quality images are collected.

At last, we need to do some data augmentation to increase the variations of the dataset. In the real-life, the common disruptions would be blocking or the difference in angles.The blocking augmentation is done during the training of BlazePose and for different angles, unfortunately, we are not able to find enough materials from internet where people are training in different camera angles, but we can simply apply horizontal flipping. I also tried using the python imgaug library to do the deformation, but as the result the performance became worse and the potential reason might be that the human body structure is destroyed during the processing. When the flipping is done, we run the BlazePose model and record those key points for the coming training stage. At last we have prepared around 100 images for each key fitness pose as at this stage, we could already have an acceptable result.

3.4. Test dataset for whole system

In order to verify the performance and effectiveness of our system, the four of us recorded a large number of exercise videos with reference to our reference video, including repeated squats for a single person at different times, and two squats at different speeds for the same number. Some of these videos will be used to verify our hybrid model, and some will be used to record the demonstration video effects.

4. PROPOSED SYSTEM

4.1. System structure

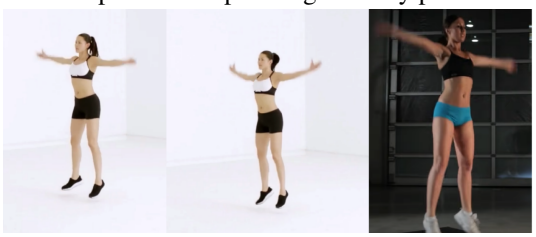
As shown in Figure 5, in the final runnable system, we transfer the input video and the image set segmented by it in the form of a stream in the model, and finally the summary is converted into a marked video by the drawer.

When we read a video, we first read it as a picture stream, and then use yolo to perform body recognition on it. The YOLO module will cut the original image stream into separate image streams containing only a single person.

At this stage, we obtain an image stream that ignores most of the background, but still includes all the features of the action. It can help our remaining models to better extract features and analyze them.

Initial sample images of key pose

Expanded sample images of key pose



Augmented sample images of key pose

Plotted sample images of key pose

Fig. 4. Pose Classification Image Sets.

Next, we need to transfer the image stream to the repnet module. The repnet module will not change the picture stream, but it can additionally obtain the repetitive information of the picture stream, that is, which frame of which cycle each image belongs to.

This information will also be provided to the video clipper, which will divide the image stream into multiple sub-image streams based on the cycle information, and each sub-image stream corresponds to a cycle.

The pose module classifies all sub-image streams, and the classification is based on the reference video we collected before. The classification will provide a score for the action at the same time, this score is our scoring of the standard degree of the action.

At the end, the drawer will draw all the results on the original video to make our results more intuitive and clear.

4.2. YOLO And Process

a)We train a YOLOv3 model by ourselves and compare with offical provided YOLOv3 model, we find offical model is better,So we decide to use offical model. We extract 50 frames from the picture set in the video and apply YOLO for target detection. If all detections are done, it will consume a lot of time and is unnecessary. Since our system supports multiple targets and single targets, when yolo detects different numbers of targets, we will also record the information of each object at the same time.

b)For a single target, we can make an accumulation and take out the largest box for the bounding box processed by yolo for resize processing. But for multiple targets, the multiple target classes generated by yolo for each frame are not fixed. We will find the corresponding bounding box set of each target for the fixed position of the target in multiple frames, and then accumulate the maximum value. Frame and resize processing to match the input of RepNet. We wrote a resize method to adapt to our requirements. We then ensure that the ratio remains the same, add black borders to the surroundings to process the picture, of course, if the length or width is greater than the given threshold, it will not Need to be added. In this way, in order to make the cropped picture set more stable, it is conducive to the analysis and use of subsequent models.

c)At the same time,Due to the performance problem of yolo, some single target detection will always identify two targets, which cannot be solved by the NMS method, so we also calculated the IOU values between different targets to filter out target boxes with a Coverage greater than 0.5.

4.3. Period estimator

In this part we introduce our period estimator architecture, the the stucture comes from Counting Out Time: Class Agnostic Video Repetition Counting in the Wild [2] which is composed

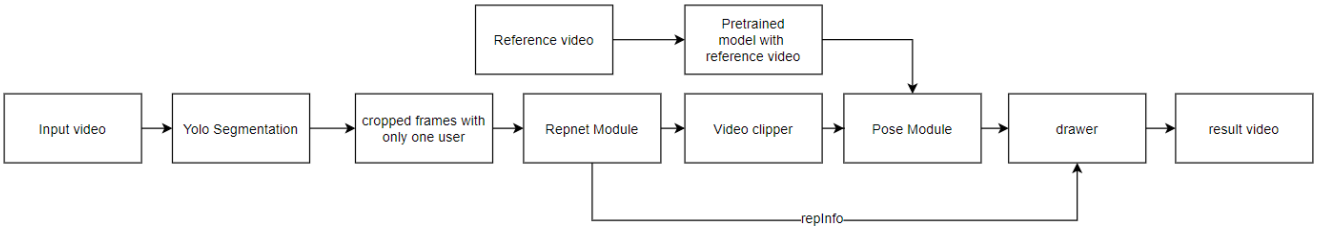


Fig. 5. system architecture.

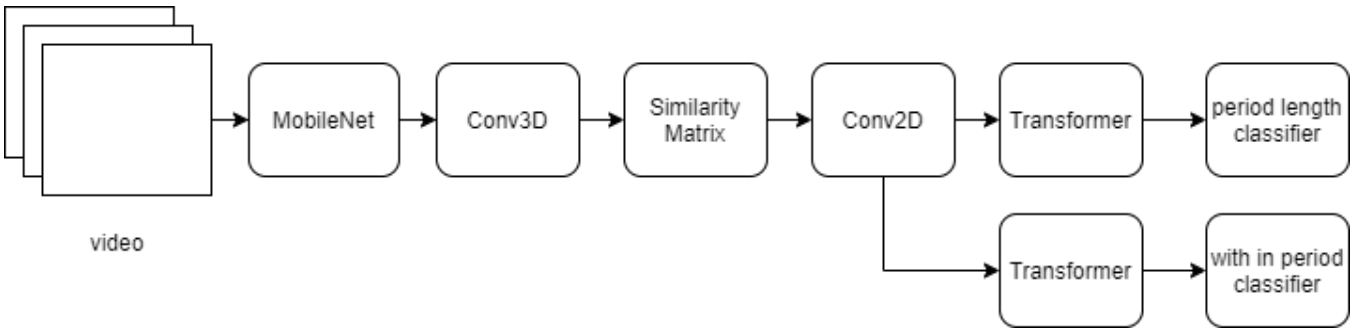


Fig. 6. **period estimator architecture.**

of two learned components, the encoder and the period predictor, with a temporal self-similarity layer in between them [2]. But the details has some differences as we want a more rapidly and easier training model.

One difference is the encoder. In our project, we want to use this model to count the motion number when doing exercise.In this case, the object is always human and the environment is always indoor. So we use pre-trained MobileNet[9] as backbone, which is usually considered in less complex environment Figure 7. The output of MobileNet is 4 X 4 X 1280 when input image size is 96 X 96. The other one is that we use two transformer layers respectively in the last two classifiers. One classifier is for predicting the period length of the frame and the other is getting the period possibility of a frame. One transformer layer shared by each classifier may improve in generalization, but it will take more time to train and tune the model. The whole structure is shown as Figure 6

4.4. Pose estimation

We used pre-trained model in mediapipe pose module. BlazePose is inspired by google's previous research BlazeFace, they made a brave assumption that in a human detection task their should always be a head that could be detected. The other body detection model usually apply NMS to get the most proper prediction area. However, this always performs unstable when human is in a movement. So they will locate the head first, and base on the head, predict the whole body by inferring the parameters like the center point around hip,

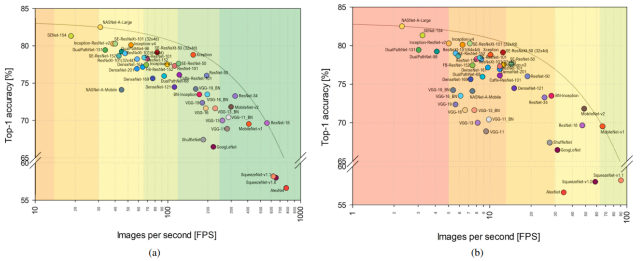


Fig. 7. Ball chart reporting the Top-1 and Top-5 accuracy vs. computational complexity.[10]

the Vitruvian man round shape of the human boundary and the tilt lean. The detection and tracking setting are shown as the Figure 8:

Compared with OpenPose and Kinect, they introduce a new structure which will apply 33 key points in total. This cooperates with their BlazeFace and BlazePalm in Coco dataset. The key points definition is shown in Figure 9.

In the neural network, they combined the heatmap, offset, and regression approach. During the training stage, only use heatmap and the offset loss, and delete the output layer from model in before running the inference. To balance the parameters, they also utilize skip-connections between all the stages of the network. This not only improve the heatmap predictions, but also significantly increase the accuracy of the coordinate regression.

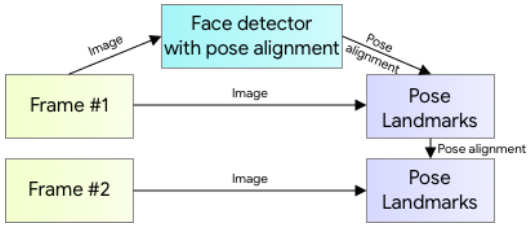


Fig. 8. BlazePose Inferring Channel.

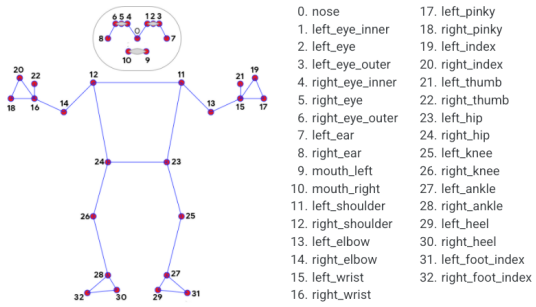


Fig. 9. Blaze Pose Key Points.

4.5. Pose classification

The k-NN algorithm used for pose classification requires a feature vector representation of each sample and a metric to compute the distance between two such vectors to find the nearest pose samples to a target one.

Pairwise distances between predefined lists of pose joints such as body parts like distances between wrist and shoulder are used as feature vectors of the pose landmarks. All poses are normalized before the conversion to adapt to the torso sizes and vertical torso orientations of different people.

To get a better classification result, k-NN search is invoked twice with different distance metrics. In the first search, minimum per-coordinate distance is used as distance metric to filter out samples that are almost the same as the target one but have only a few different values in the feature vector.Then for the second search, average per-coordinate distance is used to find the nearest pose cluster among those from the first search. Finally, exponential moving average smoothing is applied to level the noise from pose prediction or classification. In this approach, nearest pose cluster searching and probability calculation are used to do the smoothing.

In the mediapipe official site pose module, they introduce a method to classify a motion by tracking the scores of one key pose to see if it is entering or exiting the pose. However, this would not work when we apply the classification for many different fitness poses especially when there are overlapping poses, for example, the squat up actually is standing which will appear in most of the training conditions. Mean while, the on status tracking will definitely fail when there are more than two key poses in a fitness motion. To implement this and apply classification for our system, we have

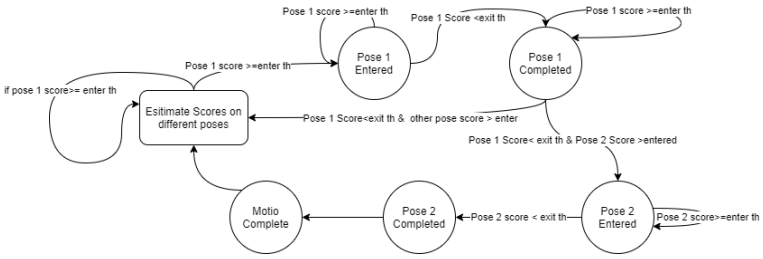


Fig. 10. FSM in two key poses motion case.

designed a new pattern for classification. Firstly we'll define a Finite State Machine(FSM) for each fitness motion. During the frames, we will estimate the scores of all the poses for each frame, and only when it follow and complete the trail of FSM we'll consider it a complete motion. The average score of the key poses then will be taken as the score of this motion. As shown in the Figure 10, this is an example of FSM pattern in two key poses case, adding more key pose just add on the pose status.

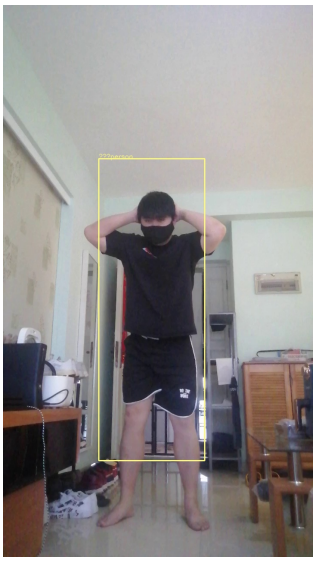
5. EXPERIMENTAL RESULTS

5.1. YOLO And Process Part

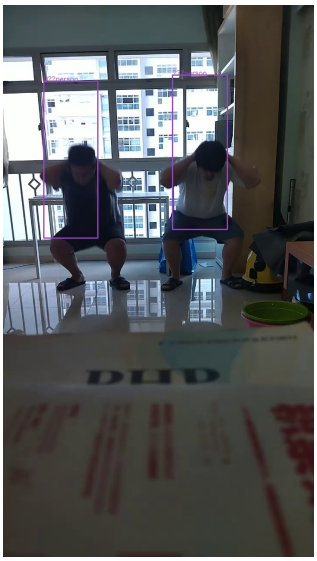
a) For the YOLO model trained by ourselves.1) Using source code: qqwweee/keras-yolo3: A Keras implementation of YOLOv3 (Tensorflow backend) and download pre-training weights and data sets. 2) Modify yolo.cfg: modify classes to 1 (3 places) filters=255 is modified to filters=18, the value formula is (classes+5)*3 Modify voc-annotation.py: modify classes to classes = ["person"]. Modify coco-class.txt and voc-class.txt in the model-data folder Only the person is left.Batch-size=8, epoch =50. The test result is shown in Figure 11. The Performance of training only 50 generations is far from the official weights, and the object box cover is poor, so the official pre-training model was finally selected.

For the YOLO pre-trained model, we defined txt as person only, Becasue in our project, we only need to detect person and crop. For the YOLO Detected method,we set Hyparameter as scFactor=1/255, nrMean=(0,0,0), RBSwap=True, scoreThres=0.8 and nmsThres=0.4. And every class a score. We set when confidence >0.5, we Assume it is the correct object bounding box. Perform non-maximal suppresion to remove redundant overlapping bounding boxes and get a single box with the highest confidence score,bboxes= boxes, scores= confidences, score threshold= scoreThres, nms-threshold= nmsThres. This time the object detect performance better. The test result is shown in Figure 12

b)For Process part, cause 256*256 image input is required for RepNet, after we apply YOLO object detection, we can tailor the video according to different target numbers.By finding the complete movement of this target in the entire video, crop the most suitable size.The single object and multiple ob-



(Single Person)



(Multiple Person)

Fig. 11. YOLO Trained by ourselves.

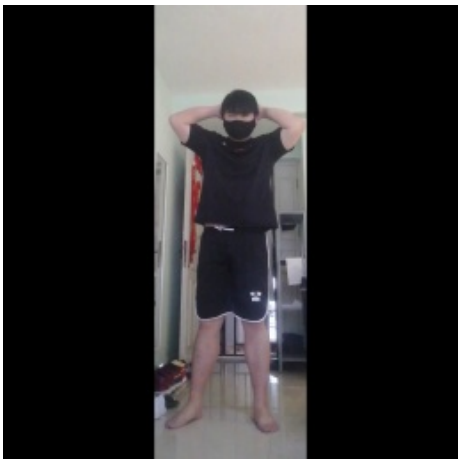
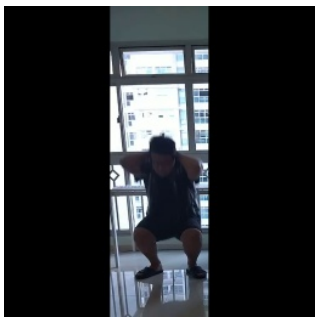
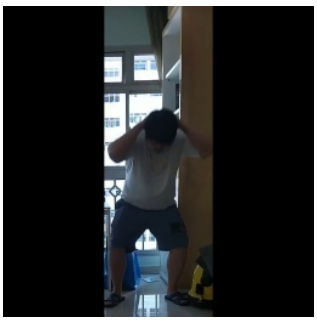


Fig. 13. Single Object.

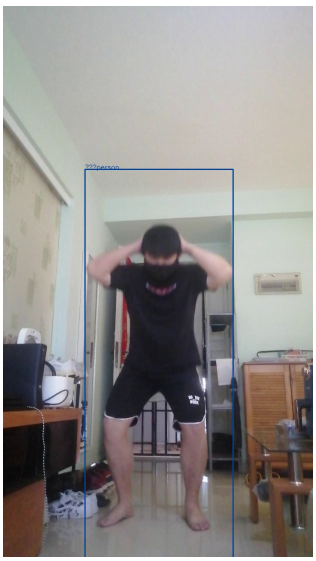


(Single Person)

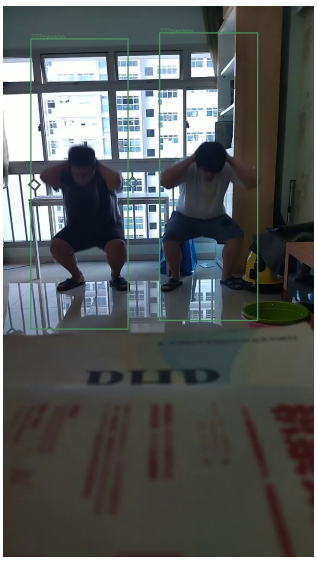


(Multiple Person)

Fig. 14. Multiple Object.



(Single Person)



(Multiple Person)

Fig. 12. YOLOV3 Pre-train model.

ject test result are shown in Figure 13 and Figure 14

5.2. Period estimator

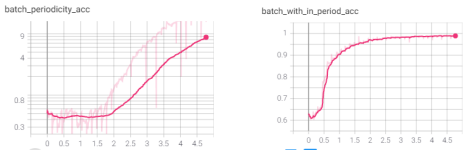
In our practice, we train the period estimator with 2 stages. In the first stage , we train the model with synthetic video without any augmentations, because we want validate the model at the first stage. And in the second stage we train this model with the random blender video. And totally we train this model with the ADAM optimizer(learn rate = 6e-6), with 110k(40k+70k) steps and batch size is 2. The whole size of the train dataset is 6k videos (64, 112, 112, 3). We still use the size of 112 here because we also train another model with ResNet-50 on this dataset.

For the 2 classifier, we apply different loss functions, binary-cross entropy for within period classifier and category-cross entropy for period length classifier. And the metrics are also different, we use binary accuracy for within period classifier, and a custom period length accuracy metric for period length classifier. In this custom period length accuracy metric function, we put the historical train data into a confusion matrix, then calculate the f1 score after every train step and return the result.Runtime accuracy is shown in Figure15 and

Further more, we select around 100 videos in Countix[8] including bench pressing,exercising arm,front raises,jumping jacks,lunge,squat .With those videos we calculate the Mean Absolute Error (MAE) of count on 3 different models: pre-trained repnet provided in [2], and 2 repnet models based on MobileNet. The result is shown in Table3

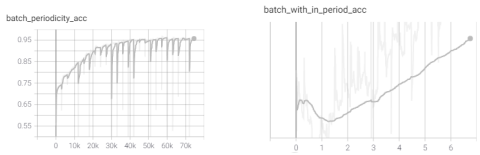
Table 3. The performance comparison.

Backbone	Parameters	MAE
ResNet50(pre-trained)	25, 673, 185	0.8961
MobileNet(no augmentation)	24, 018, 273	1.6156
MobileNet(augmentation)	24, 018, 273	0.9123



(a) (b)

Fig. 15. periodicity accuracy in 1st stage training.

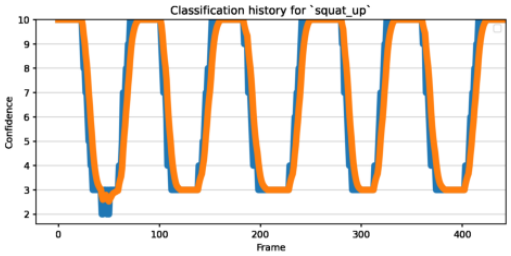


(a) (b)

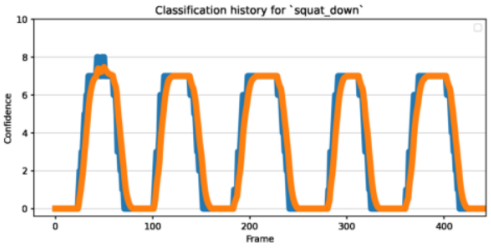
Fig. 16. periodicity accuracy in 2nd stage training.

Table 4. The performance comparison.

Model Apple Vision	BlazePose	AlphaPose	Dance(mAP)
Yoga(mAP)	62.6	63.4	32.8
Yoga(PCK@0.2)	95.5	96.0	82.7
Dance(mAP)	73.0	57.8	36.4
Dance(PCK@0.2)	97.2	95.5	91.4
HIIT(mAP)	74.0	63.4	44.5
HIIT(PCK@0.2)	97.5	96.0	88.6



Squat Up Scores



Squat Down Scores

Fig. 17. Key Poses Scores in Squat.

5.3. Pose Estimation and Classification

To evaluate the accuracy of the BlazePose, the author choose other well-performing publicly available models as baseline, and predict 17 keypoints only to be consistent. In the most recent model, they applied Yoga, Dance and HIIT as validation datasets and make a comparison with AlphaPose and Apple Vision on mAP and PCK@0.2.

It is clear to see that BlazePose is relatively more stable and doesn't have significant shortage in different occasions compared with the other models.

For the pose classification part, firstly when we get the result of the k-NN, we applied an squat video which take 5 repeats in total to see if the status changing process follows the FSM design, by plotting the scores by frames, we can visualize the status change in Figure 17:

Then we collect videos by recording by ourselves. 4 videos in total are prepared: 1. 10 Squats(Hands to chest). 2. 10 Push-Ups 3. 10 Jump-Jacks 4. 10 Wrong Squats(Hands raised) 5. 10 Random clipped video with human in. The result we get can be found in the confusion matrix in Figure 18:

The confusion matrix doesn't has the information of scores. In fact, for those failed cases, most of them has a very low score since they might perform bad in at least one fitness key pose. The bias on unknown case is caused by the situation that some random sequence are just fit in some of the fitness motion and the score is low. But this raise a issue that if for further development, we were considering more complex motions, when of overlapping of key poses increase (for example, Poppy Jump includes all there motions we pre-defined above), the method might fail at that time. However, since in real-life condition, we could ask user for particular input in a particular port, it is still possible to do by offering another model for the complex case. Then the data collection will be another trouble need to be solved.

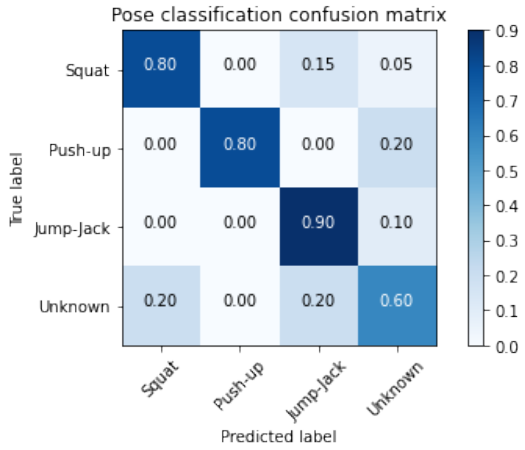


Fig. 18. Confusion Matrix.

6. CONCLUSIONS AND FUTURE WORK

In this practice module, we have successfully built a system that mixes multiple models to score the motion quality of repetitive motion videos. The system includes three parts: human body recognition, repeated segment segmentation, and action comparison. We use feature extraction, traditional graphics methods, transfer learning, model lightweighting, and update based on existing models.

If we want to update and upgrade this system in the future. We can improve from four main perspectives.

1. Using a human body segmentation model instead of YOLO, this will allow us to obtain motion information without background, which will help us greatly improve our accuracy.

2. Add more reference actions, and cooperate with professional fitness teams to record more accurate multi-angle reference action videos, so that our action comparison part can provide substantial help to the fitness group.

3. Build a real front-end and back-end system. Since our current main goal is to train the model and not to commercialize it, we used a relatively intuitive colob notebook as the final result. If we build a complete set of front-end and back-end, this will make this system capable of commercialization.

4.In current approach, we just sample the video and cut into pieces if the video is too long to predict.It should be better if apply slide window to extract frames.

7. CONTRIBUTIONS

Ding Yuxing: 1. Deployed and tested the mediapipe pose module. 2. Collected and created fitness motion data for training and testing. 3. Modified FSM and produce in code together with k-NN classification method to enable key pose based fitness motion classification.

Sun Chengyuan:I am mainly responsible for the video input of this project, the selection of the target detection model, the processing of YOLOV3 and the training of YOLOV3 model (the performance is not good and has been deprecated), and the image processing method based on single target demand and multi-target demand after target detection is completed. Deploy IOU algorithm filter duplicated box we detected to avoid the situation where multiple targets are detected for the same target due to model performance.Downflow will send the processed picture collection to Repnet for further analysis.

Wang Futong:My main work content includes model research, performance testing, hybrid model and model deployment, as well as the final integration work.

In the early stage, I studied and tried existing models of repeated video segmentation, including repnet, and compared their feasibility, stability, training difficulty, and actual running speed.Then I helped my teammate to further research and test the segmentation of yolo, and helped him complete the YOLO module.

In the subsequent model fusion and deployment phase, we hope to use colab notebook to display our model in a file. This work is done by me. I studied how to adjust the output of a model so that it can be the input of another model or the input of a video splitter. I wrote a structure that is as clear as possible. When my teammates make any changes to their part of the model, we can easily modify the corresponding part without affecting others.

Xu Mingjie: 1. Collect video data from YouTube for the period estimator training, and also create functions for synthetic video generation as well as augmentation. 2.Repnet[2] implementation and tune with different backbones. 3.Create functions for video clipping

8. REFERENCES

[1] Joseph Redmon and Ali Farhadi, "Yolov3: An incremental improvement," 2018.

[2] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman, "Counting out time: Class agnostic video repetition counting in the wild," 2020.

[3] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7291–7299.

[4] Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, and Matthias Grundmann, "Blazepose: On-device real-time body pose tracking," *arXiv preprint arXiv:2006.10204*, 2020.

[5] Huijuan Xu, Abir Das, and Kate Saenko, "R-c3d: Region convolutional 3d network for temporal activity detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5783–5792.

[6] Sophie Aubry, Sohaib Laraba, Joëlle Tilmanne, and Thierry Dutoit, "Action recognition based on 2d skeletons extracted from rgb videos," in *MATEC Web of Conferences*. EDP Sciences, 2019, vol. 277, p. 02034.

[7] Tom F H Runia, Cees G M Snoek, and Arnold W M Smeulders, "Real-world repetition estimation by div, grad and curl," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[8] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman, "The kinetics human action video dataset," *arXiv preprint arXiv:1705.06950*, 2017.

[9] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017.

[10] Simone Bianco, Remi Cadene, Luigi Celona, and Paolo Napoletano, "Benchmark analysis of representative deep neural network architectures," 2018.