Evaluating pose estimation and object detection models for the application in the minisoccerbal project

Parviz Ahmadov, University of Twente, The Netherlands

The focus of this research is to evaluate several lower extremity and sports ball detection models to determine whether the combination of the two can later be used to calculate several parameters of the exercises that are performed with a soccer training object called minisoccerbal. There are already a number of existing machine learning libraries and algorithms that are able to detect lower body joints and a sports ball in an image. This research aims to examine existing tools' effectiveness in detecting the required objects in videos that are within constraints relevant to the MiniSoccerball project, that is, the use of a mini soccer ball, stationary camera, etc. Aside from the accuracy of algorithms, processing speed is also a priority, with real-time object detection being the future direction. Two object detection models (for minisoccerbal detection) - YOLOv5 and EfficientDet, and two pose estimation models - OpenPose and BlazePose (for lower body joint detection) were chosen for evaluation. mAp (Mean Average Precision) scores were used for evaluating object detection models while Pose estimation models were evaluated based on PDJ scores (Percentage of Detected Joints). FPS (Frame per second) was calculated for determining the processing speed for all the models. Although EfficientDet had a slightly higher mAP score compared to YOLOv5, YOLOv5 was chosen as the more suitable model because of the speed advantage and having a sufficiently high mAp@0.75 score. For pose estimation, OpenPose was determined to be more suitable despite being significantly slower, due to BlazePose having a lower PDJ score.

1 INTRODUCTION

Minisoccerball is a soccer ball that has a cord attached to it with the other side of the cord meant to be connected to the player. This ensures that the ball never leaves the side of the player, making several soccer exercises possible along with allowing more movements in a short duration. The product is mainly aimed at young players in the age range of 6 to 12 and is meant to help them practice control over the ball. According to the research by Fay Zhang [7] the coaches would be interested to obtain information about exercise parameters, such as a number of ball contacts, speed of the ball, and ratio of left to right contact when the students perform the exercise for them to review the performance. To accomplish this, the method of embedding sensors on the ball to detect touches and calculate parameters has been explored before [22]. However, changing the structure of the ball by embedding it with sensors and a battery impacted the integrity of the ball, therefore the idea has been dismissed. With the current project, the objective is to achieve this through the use of object detection to determine the coordinates of the ball and the lower body and calculate numerous parameters based on them. This research aims at testing existing tools' effectiveness for the minisocerbal project. The purpose is to successfully detect human lower body joints and a mini soccer ball under the constraints of the project - when the camera is stationary, the minisocerbal is used for performing the exercises, the distance between the ball and the camera is not higher than a few meters. For this purpose, the most suitable models will be selected and evaluated. The paper ultimately aims to contribute to the ongoing research on detecting sports equipment and players.

TScIT 37, July 8, 2022, Enschede, The Netherlands

Although research about player and sports equipment detection has been done before [2, 11, 22, 25], there is no existing research that compared the existing models to propose the most effective ones for calculation of exercise parameters with the minisoccerbal. Therefore, the aim is to review the various object detection and pose estimation models to help identify the mini soccer ball and the lower body joints of soccer players. Results will be evaluated using a relevant evaluation metric to determine the ideal model with a balance between accuracy and speed. Since pose estimation models detect various keypoints and sports ball detection models output bounding box, different evaluation strategies will be used for each of them. Following research questions have been developed in order to accomplish the goal:

- Which existing object detection and pose estimation models are the most relevant for tracking the lower body joints and soccer ball location in a video stream?
- How do the choice of an exercise and the distance affect the accuracy of the models?
- What combination of models can be considered the most suitable for the task based on accuracy and processing speed?

2 RELATED WORK

There have been considerable advancements in the field of object detection over the past several years. The first solution to the problem of object detection that involved deep neural networks was proposed in 2014 which utilized the sliding window approach [21]. This approach had a very high computation time, as not only did same-sized bounding boxes have to be fed into the convolutional neural network after each slide, but the size of the bounding boxes also had to be modified to account for different-sized objects in an image. Shortly after, the RCNN model was proposed by Ross Girshick which bypassed the problem of a huge number of region proposals [9]. It applied a selective search algorithm that limited the number of proposed regions to 2000, decreasing the overall computation time. In 2016, a faster RCNN model was developed [20] which offered considerably higher speed and accuracy compared to its predecessor. Despite the improvements, the faster RCNN model is still slow and requires a strong GPU for running.

YOLO is an entirely different approach to object detection and is magnitudes of time faster than its predecessors [18]. It is a One-Stage Object Detection Model meaning that it skips the region proposal step altogether and the entire image is fed into the neural network to determine the bounding box. There are many versions of the YOLO, such as YOLOv3 [19], YOLOv4 [5], YOLOv5 [30], and scaled-YOLOv4 [31] publishers of each version claiming to offer advantages over its predecessors. Currently, the models built on top of one-stage detectors such as YOLO and SSD (Single Shot Detector) [11] architecture are one of the industry leaders in terms of speed and accuracy.

Most of the single-stage detectors still require powerful GPUs for real-time object detection. Real-time ball detection is essential in RoboCup, therefore several papers have been produced that proposed new models for soccer ball detection which are faster and sufficiently accurate[12, 24, 27]. Work by Meisam Teimouri et al. [27] had good results, achieving close to 90% accuracy and precision, and was able to run in real-time. Research by Matija Buric et al. [2] evaluates Mask-RCNN and YOLO models in handball detection based on speed and accuracy.

Object detection models use a Convolutional Neural Network backbone for feature extraction to detect objects in an image. For instance, for popular models such as YOLO, SSD, and Faster RCNN there is an option to use different types of CNNs such as Inception, Resnet, and VGG for feature extraction, each of them offering different speed and accuracy. There is an option to train these backbones from scratch with randomly initialized weights or to use the ones pre-trained on public datasets, most commonly the COCO dataset. COCO dataset is large-scale object detection, segmentation, and a captioning dataset containing more than 300,000 images [4]. It has 80 object categories, one of which is sports ball, therefore, for the task of mini soccer ball detection, there is no need to train the network from scratch. One option is to use transfer learning, where the last layers of the CNN are additionally trained on a new dataset which Matija Buric et al. [2] used for training the handball detection model, however, for this research, testing will be made on the pre-trained models and no transfer learning will be used.

Human Pose Estimation models work by returning coordinates of the several important joints of the person on an input image, the output being similar to the skeleton of the human. The output coordinates are usually further processed for computing parameters for the specific application. In the case of soccer exercise analysis, the results will be used to calculate angles between parts of the human leg, distance from the ball, and speed of the joints. and determine its interactions of it with the ball. The first CNN-based human pose estimation model was proposed by Toshev et al. in 2014 called DeepPose: Human Pose Estimation via Deep Neural Networks [29]. Many pose estimation models have been developed since which utilize CNN such as OpenPose [3], AlphaPose [6], DeepCut [16], etc.

The accuracy of the models is usually determined by testing on public datasets, such as COCO test-dev or MPII Human Pose. For instance, on COCO test-dev Alphapose scored a higher Average Precision score compared to OpenPose and Mask RCNN in which models were tested based on 17 body keypoints [15]. Frederick Zhang et al. compared the performance of OpenPose and HyperPose for the task of clinical assessment [32]. Each model was evaluated qualitatively with the 4-point scale and the results showed that OpenPose was significantly more accurate. Sarah Mroz et al. compared the performance of BlazePose and OpenPose for a similar purpose and used Mean Square Metric and Pearson Correlation for comparison. Results showed that while BlazePose was significantly faster, OpenPose was more accurate [13].

In general, detection of lower body joints is a less challenging task compared to sports-ball detection due to the identifiable shape and size of the human, especially when only a single person has to be detected in the image.

3 METHODOLOGY

3.1 Choice of models

For mini soccer ball detection, YOLOv5 and Google's EfficientDet models [26] will be tested. Figure 1 shows that scaled YOLOv4 and EfficientDet models reached mAp values above 50 in the COCO dataset, higher than other popular models.

YOLOv5 was introduced as an improvement over YOLOv4 in June 2020. Authors of the YOLOv5 claim to have a mAp score slightly above EfficientDet while being significantly faster [19]. In the work done by Zheng Ge et al. YOLOv5 was found to be more accurate than YOLOv4 and YOLOv3 while having a similar inference time [8]. Upesh Nepal et al. also found YOLOv5 to be more accurate than YOLOv4 for the task of Autonomous Landing Spot Detection in Faulty UAVs [14]. EfficientDet and YOLOv5 were chosen for

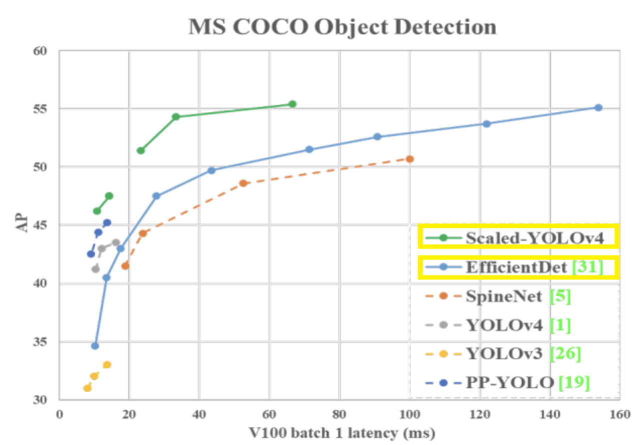
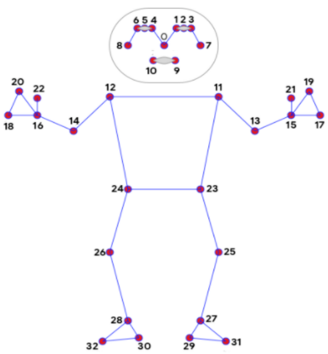


Figure 1: Comparison of the several state-of-the-art object detectors [31]

evaluation, two very accurate object detection models with different architecture for testing.

For pose estimation, OpenPose and BlazePose [1] were chosen for evaluation. Compared to most of the other pose estimation models, both of these models are able to detect the foot and heel joints of the human leg (See Figure 2 for pose landmarks detectable by BlazePose) which is the requirement for minisoccerbal exercise analysis at later stages.



0. nose
1. left_eye_inner
2. left_eye
3. left_eye_outer
4. right_eye_inner
5. right_eye
6. right_eye_outer
7. left_ear
8. right_ear
9. mouth_left
10. mouth_right
11. left_shoulder
12. right_shoulder
13. left_elbow
14. right_elbow
15. left_wrist
16. right_wrist
17. left_pinky
18. right_pinky
19. left_index
20. right_index
21. left_thumb
22. right_thumb
23. left_hip
24. right_hip
25. left_knee
26. right_knee
27. left_ankle
28. right_ankle
29. left_heel
30. right_heel
31. left_foot_index
32. right_foot_index

Figure 2: Pose landmarks in Blazepose [17]:

In our research, both models will be tested on videos where exercises with the minisoccerbal are performed. Whether the lower speed of BlazePose in favor of faster inference time is sufficient for the task or whether there is a need for a more accurate model - OpenPose will be determined.

3.2 Dataset

Dataset was obtained in collaboration with Twente Football School, where the videos of the football coach performing exercises with the minisoccerbal were recorded. The exercises were performed outside in the daytime, to account for the brightness of the background and to better represent the use case scenario. Three exercises that have the potential to be challenging for pose estimation and sports ball detection models were selected on purpose to determine how the models perform under difficult circumstances. The participant was asked to perform the exercises for a few seconds, while the camera was put on the chair in a stationary position. The participant was asked to perform the same exercises 2.5 meters and 5 meters away from the camera. Overall 6 combinations were recorded (3 exercises and 2 distances). The idea was to test the models based on 2 variables - performed exercise and the distance. Later the video was converted into frames and one in every five frames was included in the testing test. Overall, the number of frames in the testing dataset was 192 (32 per combination). The 10 lower body joints (toe, heel, ankle, knee, and hip for each leg) were annotated on Label Studio [10] and the minisoccerbal was annotated through the method of drawing a bounding box around it with the software labelimg [21]. The face of the participant will be blurred before storing the dataset to protect the anonymity of the participant.

3.3 Evaluation

Separate metrics will be calculated for the whole dataset and for each exercise and distance. For both pose estimation models and sports ball detection models, FPS (frame per second) will be calculated to evaluate the speed.

Accuracy metrics for Pose Estimation Models: PDJ (Percentage of Detected Joints) will be used as an evaluation metric. It sets the threshold value a 5 percent of the diagonal length of the bounding box of the person object. Euclidian distances between true and predicted keypoints below the threshold are considered as detected, undetected otherwise [25].

$$PDJ = \frac{\sum_{i=1}^{m} \frac{\sum_{u=1}^{n} bool(d_i < 0.05 * diagonal)}{n}}{m}$$

- m - Number of frames in the chosen dataset
- di – the euclidian distance between ground truth and the predicted keypoint,
- n – number of keypoints in an image
- diagonal - diagonal length of the bounding box of the person object in the initial frame

Besides, PDJ will be calculated per joint to determine which lower body joints are detected the best and the worst. We will call this PDJ_X where X denotes the name of the joint and $d_{(X,i)}$ denotes the euclidian distance between ground truth and predicted keypoint for Joint X in frame i.

$$PDJ_X = \frac{\sum_{i=1}^{m} bool\left(d_{(X,i)} < 0.05 * diagonal\right)}{m}$$

Accuracy metrics for sports ball detection models: For evaluating the accuracy of the models, mAp, and mAp@0.75, values will be calculated. mAp score returns a value between 0 an 1, indicating how accurate the model is in its detections. A high number of false positives and false negatives would yield a low mAp score, while mAp score would be equal to 1 if number of false negatives and false positives would be 0. mAp@x denotes mAp score when the IoU (Intersection over Union) threshold is equal to x. Higher IoU threshold indicates that bounding boxes have to match more perfectly to the true bounding box for object to be considered as detected. mAp score is calculated through averaging mAp@x scores where x increases by 0.05 in every iteration starting from the initial value of 0.5 to 0.95. This value gives a more complete idea about the detection accuracy of the model while scores mAp@0.75 helps understand what the good threshold is to choose for the final product.

3.4 Setup

The videos were shot on an iPhone 12 smartphone with a resolution of 480 x 848.

The models were tested in Python language. Both YOLOv5 and EfficientDet were tested in their development environment - the former was implemented in Pytorch while the latter was implemented in Tensorflow. The heaviest version of the YOLOv5 - YOLOv5x model was used because it has a high compensation of accuracy over a slight increase in inference time compared to lighter models. Models were run on Google Colab with the 12GB NVIDIA Tesla K80 GPU. For the calculation of mAp score, the PyMetrics library was used in the case of YOLOv5. Tensorflow's in-built metrics evaluation functionality was used for EfficientDet. A maximum number of people was set to one in OpenPose (Mediapipe only detects a single person by default). Since the computation of PDJ value requires the calculation of the diameter of the bounding box of the person object to determine the threshold, the YOLOv5 model was combined with the pose estimation models to easily determine the bounding box of the person object for the initial frame. Choosing EfficientDet for this task would most likely not make any difference, since person detection is highly accurate in both these models and since these models are used only on the initial frame. For the calculation of FPS, the processing time of the models was calculated using Python's timeit library [26] and was divided by the number of frames to determine FPS.

4 RESULTS AND DISCUSSION

The whole dataset was fed into each model and a visual representation of the results was saved. Figure 3 displays the results on one of the frames of the test dataset.

Figure 3: Detection visualization of models: A - BlazePose, B - OpenPose, C - EfficientDet, D - YOLOv5

Figure 4: Example frames where YOLOv5 and EfficientDet struggled to detect the ball due to white background

Figure 5: Example frames where BlazePose falsely detected lower body joints

4.1 Minisoccerbal detection

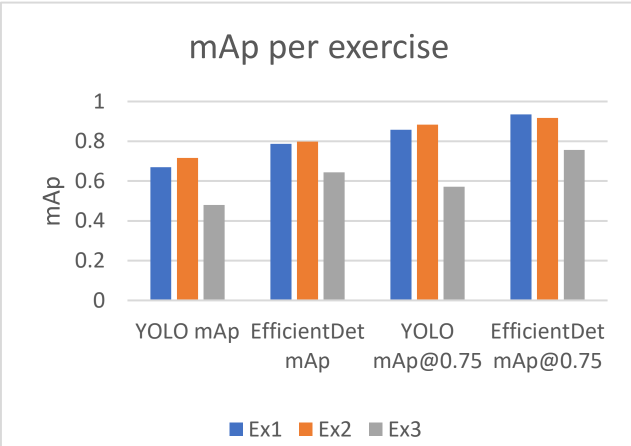


Figure 6: mAp and mAp@0.75 score per exercise

This section discusses the comparison of sports ball detection models after the calculation of the mAp score for the models with different datasets used. Figure 6 shows the side-by-side comparison of YOLOv5 and EfficientDet in terms of mAp and mAp@0.75 scores for each of the 3 exercises. As observable from the chart, EfficientDet slightly outperformed YOLOv5 in the task of mini soccer ball detection. Both models struggled to accurately detect the minisoccerbal in exercise 3. That is because the ball is flown through the top of the head, resulting in the ball being in the sky background. Since both sky and the ball are whitish color, the models struggled to detect the ball under the circumstances. As shown in Figure 4, The models also struggled to detect the ball when it touched the white snickers of the participant, for similar reasons.

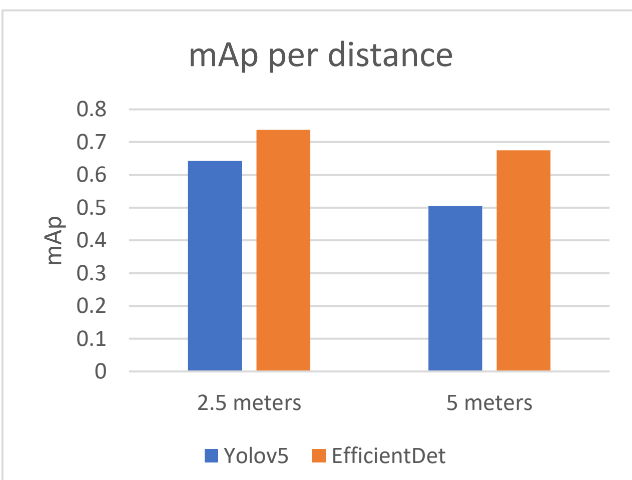


Figure 7: mAp score per distance

As noticeable from Figure 7, both models performed worse when the distance between the camera and the ball increased.

4.2 Lower body joint estimation

This section discusses the performance of the Pose detection libraries, based on the calculated PDJ score. Figure 8 shows the comparison of BlazePose and OpenPose based on PDJ per exercise. As visible from the graph, OpenPose had a higher PDJ value

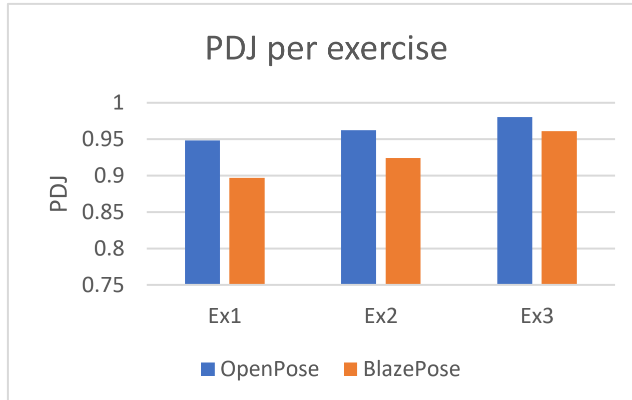


Figure 8: PDJ score per exercise

compared to BlazePose for all 3 exercises.

Both models had the lowest PDJ score for exercise 1, due to the crossing of legs in the exercise which makes it challenging for models to detect lower body joints. However, OpenPose still performed decently in the exercise, reaching a PDJ value close to 0.95. Figure 5 shows an example of a frame from exercise 1 where BlazePose falsely detected the lower body joints due to the crossing of the legs, while OpenPose detected them accurately. Models performed the best in exercise 3 because legs have the least complicated trajectory compared to other exercises.

As observable from Figure 9, the models had the lowest PDJ for the detection of the toe and heel joints. That is due to the fact that these joints constantly change their location and are frequently not visible in the image due to the ball or the other leg crossing them. Knee and hip joints on the other hand had the highest PDJ score, due to them having a less challenging trajectory in the images and being fully visible in almost all images.

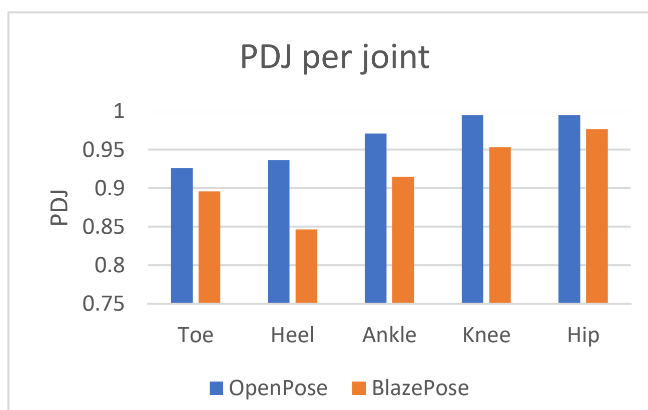


Figure 9: PDJ score per joint

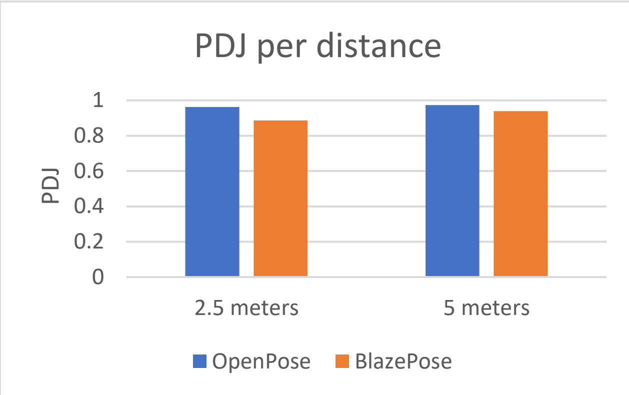


Figure 10: PDJ score per distance

As displayed on Figure 10, the distance between the ball and the camera did not have an impact on the detection accuracy in the case of pose estimation modes.

4.3 FPS

Figure 11 shows the calculated FPS values for all the models. YOLOv5 and OpenPose were significantly faster than EfficientDet and OpenPose, reaching close to 20 FPS.

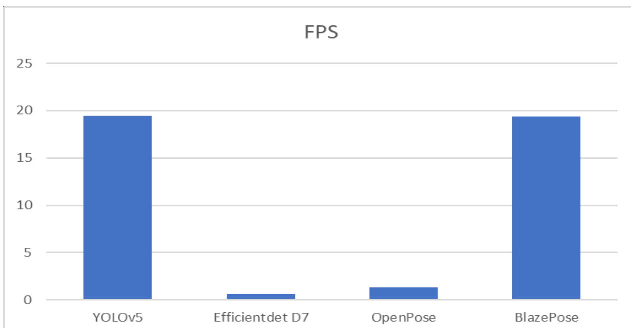


Figure 11: FPS per model

4.4 Discussion

Although EfficientDet had a slightly higher mAp score than YOLOv5 for all the exercises, both models scored above 0.85 in mAp@0.75 scores (except for exercise 3). Relatively low mAp scores are likely due to slight inaccuracies in the annotations rather than inaccuracy in detections since mAp@0.75 values are significantly higher. Based on this, it can be suggested that YOLOv5 is a better choice taking its comparable mAp and very high processing speed into consideration. Both YOLOv5 and EfficientDet struggled with the background that has a similar color to the ball, and this became apparent, especially in exercise 3. The solution to this would be to choose a ball with a high contrasting color (eg. orange). Transfer learning can also be used to increase the accuracy of the models under these circumstances.

BlazePose was unable to detect close to 10 percent of the toe joints and close to 15 percent of the heel joints in the whole dataset. Since these are the core joints involved in the ball contact, the analysis of the exercise can be more challenging with this number of undetected joints. OpenPose on the other hand, had a PDJ value close to 0.95 across all joints, and across all 3 exercises, making it a more suitable model for mini soccer ball exercise analysis despite BlazePose's high FPS score. Since the best-case scenario would be real-time detection, further investigation of fast and lightweight pose estimation models is needed.

The distance did not have an impact on the performance of pose estimation models, likely due to identifiable shape of the person. For sports ball detection models, however, distance had a negative impact on the accuracy. This implies that for the final product, it would be a good idea to ask the participant to perform the exercises close to the camera.

5 CONCLUSION AND FUTURE WORKS

The aim of the research was to evaluate object detection and pose estimation models to determine whether the exercise analysis with the minisoccerbal could be achieved and determine the suitable models to accomplish that. Overall 4 models were chosen and were evaluated based on appropriate metrics on the obtained dataset. After calculating accuracy scores on different exercises, distances, and joints and analyzing the results, it can be concluded that the task is achievable, with the ideal combination being YOLOv5 and OpenPose. However, there are a few issues with the models, and some factors to take into consideration were discussed in the earlier section.

One of the limitations of the paper was that the dataset was tested based on pre-trained weights on the COCO dataset and was not additionally trained. Transfer learning could be used, where the pre-trained model is trained additionally with the custom training dataset to better adjust for minisoccerbal product. Besides, rather than testing the processing speed of the combination of the pose estimation and sports ball detection models, they were tested individually. Analyzing the models in combination would most likely yield more reliable data for the evaluation.

Future work might look into combining the chosen models and using the output coordinates to calculate exercise parameters. Fay Zhang, who is doing similar research [7] provided the list of important parameters that are necessary for minisoccerbal exercise analysis after an interview with the coach.

- Ball contacts
- Speed
- Ball rotation
- Ratio of left to right contact
- Pattern of kicking in the tie frame: LR RL LL LR

Calculation of these metrics could be achievable using the output coordinates of the models, except for the ball rotation which could be more challenging to determine. Ball contacts, for example, could be computed by determining the frames with the minimum distance between the ball and the feet, and when the ball changes its direction. The to attempt calculate these parameters using the data our research provides would be a meaningful continuation to the ongoing research on minisoccerbal.

5 REFERENCES

1. Bazarevsky, V., Grishchenko, I., Raveendran, K., Zhu, T., Zhang, 12. F., & Grundmann, M. (2020, June 17). *Blazepose: On-device real-time body pose tracking*. arXiv.org. Retrieved June 26, 2022, from https://arxiv.org/abs/2006.10204

2. Buric, M., Ivasic-Kos, M., & Pobar, M. (n.d.). Ball detection using Yolo and mask R-CNN. IEEE Xplore. Retrieved May 8, 2022, from https://ieeexplore.ieee.org/abstract/document/8947818/authors#authors

3. Cao, Z., Hidalgo, G., Simon, T., Wei, S.-E., & Sheikh, Y. (2019, May 30). OpenPose: Realtime multi-person 2D pose estimation using part affinity fields. arXiv.org. Retrieved June 26, 2022, from https://arxiv.org/abs/1812.08008

4. Common objects in context. COCO. (n.d.). Retrieved June 26, 2022, from https://cocodataset.org/#home

5. Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., & Tian, Q. (1970, January 1). CenterNet: Keypoint Triplets for object detection. CVF Open Access. Retrieved June 25, 2022, from https://openaccess.thecvf.com/content_ICCV_2019/html/Duan_CenterNet_Keypoint_Triplets_for_Object_Detection_ICCV_2019_paper.html

6. Fang, H.-S., Xie, S., Tai, Y.-W., & Lu, C. (2018, February 4). RMPE: Regional Multi-person pose estimation. arXiv.org. Retrieved June 26, 2022, from https://arxiv.org/abs/1612.00137

7. Fay, Zhang (2022) Dashboard Design for MiniSoccerbal 3.0 [Bachelor thesis, Twente University]. SDI Project data repository

8. Ge, Z., Liu, S., Wang, F., Li, Z., & Sun, J. (2021, August 6). Yolox: Exceeding Yolo Series in 2021. arXiv.org. Retrieved June 26, 2022, from https://arxiv.org/abs/2107.08430

9. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014, October 22). Rich feature hierarchies for accurate object detection and semantic segmentation. arXiv.org. Retrieved May 5, 2022, from https://arxiv.org/abs/1311.2524

10. Label studio ‑ open source data labeling. Label Studio ‑ Open Source Data Labeling. (2022). Retrieved July 3, 2022, from https://labelstud.io/

11. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (1970, January 1). SSD: Single shot multibox detector. SpringerLink. Retrieved June 25, 2022, from https://link.springer.com/chapter/10.1007/978-3-319-464480_2

12. Menashe, J., Kelle, J., Genter, K., Hanna, J., Liebman, E., Narvekar, S., Zhang, R., & Stone, P. (1970, January 1). Fast and precise black and white ball detection for RoboCup soccer. SpringerLink. Retrieved May 8, 2022, from https://link.springer.com/chapter/10.1007/978-3-030-003081_4

13. Mroz, S., Baddour, N., McGuirk, C., Tu, A., Cheung, K., & Lemarie, E. (2022, January 21). Comparing the quality of human pose estimation with Blazepose or openpose. IEEE Xplore. Retrieved June 26, 2022, from https://ieeexplore.ieee.org/abstract/document/9677850?casa_token=BVa8Wc7WCq4AAAAA%3AokcQhs8gxIxx8v59wk241GUwvSeasg5mI_7bmzh1MEiO3mYv2x IWweAJjPee1RbTGAAbchFROY

14. Nepal, U., & Eslamiat, H. (2022, January 8). Comparing Yolov3, Yolov4 and Yolov5 for autonomous landing spot detection in faulty uavs. MDPI. Retrieved June 26, 2022, from https://www.mdpi.com/1424-8220/22/2/464

15. Papers with code - coco test-dev benchmark (pose estimation). The latest in Machine Learning. (n.d.). Retrieved June 26, 2022, from https://paperswithcode.com/sota/pose-estimation-oncoco-test-dev

16. Pishchulin, L., Insafutdinov, E., Tang, S., Andres, B., Andriluka, M., Gehler, P., & Schiele, B. (2016, April 26). Deepcut: Joint subset partition and labeling for multi person pose estimation. arXiv.org. Retrieved June 26, 2022, from https://arxiv.org/abs/1511.06645

17. Pose. mediapipe. (n.d.). Retrieved May 5, 2022, from https://google.github.io/mediapipe/solutions/pose

18. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016, May 9). You only look once: Unified, real-time object detection. arXiv.org. Retrieved May 5, 2022, from https://arxiv.org/abs/1506.02640

19. Redmon, J., & Farhadi, A. (2018, April 8). Yolov3: An incremental improvement. arXiv.org. Retrieved June 25, 2022, from https://arxiv.org/abs/1804.02767

20. Ren, S., He, K., Girshick, R., & Sun, J. (2016, January 6). Faster R-CNN: Towards real-time object detection with region ... - arxiv. Retrieved May 5, 2022, from https://arxiv.org/pdf/1506.01497.pdf

21. Russel, Bryan et al. (2013) LabelImg [Source code]. https://github.com/tzutalin/labelImg

22. R. van Weering (2021) MiniSoccerbal 2.0: An instrumented Minisoccerbal to trackperformance, improve technique and give feedbackto soccer players [Bachelor thesis, Twente University]. SDI Project data repository

23. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2014, February 24). Overfeat: Integrated recognition, localization and detection using convolutional networks. arXiv.org. Retrieved May 5, 2022, from https://arxiv.org/abs/1312.6229

24. Speck, D., Barros, P., Weber, C., & Wermter, S. (1970, January 1). Ball localization for Robocup soccer using Convolutional Neural Networks. SpringerLink. Retrieved May 8, 2022, from https://link.springer.com/chapter/10.1007/978-3-319-687926_2

25. Stasiuk, A. (2020, November 11). Pose estimation. metrics. Medium. Retrieved July 3, 2022, from https://stasiuk.medium.com/pose-estimation-metrics844c07ba0a78

26. Tan, M., Pang, R., & Le, Q. V. (2020, July 27). EfficientDet: Scalable and efficient object detection. arXiv.org. Retrieved June 25, 2022, from https://arxiv.org/abs/1911.09070

27. Teimouri, M., Delavaran, M. H., & Rezaei, M. (1970, January 1). A real-time ball detection approach using convolutional neural networks. SpringerLink. Retrieved May 6, 2022, from https://link.springer.com/chapter/10.1007/978-3-030-356996_25#Fn1

28. Timeit - measure execution time of small code snippets ¶. timeit - Measure execution time of small code snippets - Python 3.10.5 documentation. (n.d.). Retrieved July 3, 2022, from https://docs.python.org/3/library/timeit.html

29. Toshev, A., & Szegedy, C. (1970, January 1). DeepPose: Human pose estimation via Deep Neural Networks. CVF Open Access. Retrieved June 26, 2022, from https://openaccess.thecvf.com/content_cvpr_2014/html/Toshev_DeepPose_Human_Pose_2014_CVPR_paper.html

30. ultralytics (2020) yolov5 [Source code] https://github.com/ultralytics/yolov5

31. Wang, C.-Y., Bochkovskiy2, A., & Liao, H.-Y. M. (2021, June). Scaled-yolov4: Scaling cross stage partial network. CVF Open Access. Retrieved June 25, 2022, from https://openaccess.thecvf.com/content/CVPR2021/papers/Wang_ScaledYOLOv4_Scaling_Cross_Stage_Partial_Network_CVPR_2021_ paper.pdf

32. Zhang, F., Lemarie, E., Baddour, N., Cheung, K., Tu, A., McGuirk, C., & Juneau, P. (2021, July 12). Comparison of openpose and HyperPose artificial intelligence models for analysis of handheld smartphone videos. IEEE Xplore. Retrieved June 26, 2022, from https://ieeexplore.ieee.org/abstract/document/9478740