Landing Error Scoring System for Basketball: A Computer Vision Approach

Dhairya Shah
Btech. CSE
School of Engineering and Applied Sciences
Ahmedabad, India
dhairya.s4@ahduni.edu.in

Raj Dave
Btech. CSE
School of Engineering and Applied Sciences
Ahmedabad, India
raj.d@ahduni.edu.in

Aayushi Shah
Btech. CSE
School of Engineering and Applied Sciences
Ahmedabad, India
aayushi.s3@ahduni.edu.in

Vanaja Agarwal
Btech. CSE
School of Engineering and Applied Sciences
Ahmedabad, India
vanaja.a@ahduni.edu.in

***Abstract*—The Landing Error Scoring System (LESS) method is used in sports, but scoring requires a lot of time, depends on the clinician, and is typically unavailable to athletes in lower divisions of competition. An objective, non-invasive, and time-efficient way to evaluate jump-landings and determine damage risks is through the use of computer vision techniques. Our aim is to automate those LESS scores using a deep-learning-based computer vision approach. We use the countermovement jump video of one of the athletes out of the 17 NCAA Division I female basketball athletes to identify the potential injury risks, at the same time we are taking a step towards automating the repetitive task of calculating the LESS scores and detecting errors.**

***Index Terms*—image processing, computer vision, LESS, athletes, basketball, injury**

I. Introduction

Sports with high-intensity, repetitive and asymmetric postures tend to increase the likelihood of injuries [2]. Jump landings are one of the many movements which are associated with athletes injury risks. Evaluating the jump landings and scoring them based on a standardized scoring system (LESS) to assess injury risk becomes a valuable tool. The values of knee flexion, stance width, internal/external rotation are major predictors of anterior cruciate ligament (ACL) injury risk [3]. Furthermore, lateral trunk flexion for an athlete becomes an important tool to measure the readiness of the athlete to participate in the sport after an injury [4].The extreme labor-intensive process of physically assessing jump landing scores through LESS, can be simplified by using Machine Learning (ML) and Computer Vision (CV) [5]. We consider the two primary metrics for injury risk through LESS, Lateral Trunk Flexion and Stance Width. Through consistent data collection through video frames fragmented into individual frames, we standardize the method to calculate the respective error score through the computer vision methods. This would in turn help develop a generalized method to automate the error detection in Jump Landing videos.

Previous works have focused on calculating the error of each frame through the key points detected via a Human Pose Estimation algorithm and relied on those calculated values for the scores. However, our aim is to eliminate the overhead caused by processing the key points for each frame and calculating the errors for each frame. Thus, we propose to use a deep learning model tuned for Jump landing videos which skips the key point extraction and directly classifies if the error is present in the frame or not along with information about the type of error (e.g. right_lateral_flexion, narrow_stance, etc.) For instance, the work from [1] relies upon the extraction of key points through MediaPipe Pose and then calculate the error scores, instead we propose an automated system which uses less computational resources for the same task.

II. Methodology

A. Frame Extraction

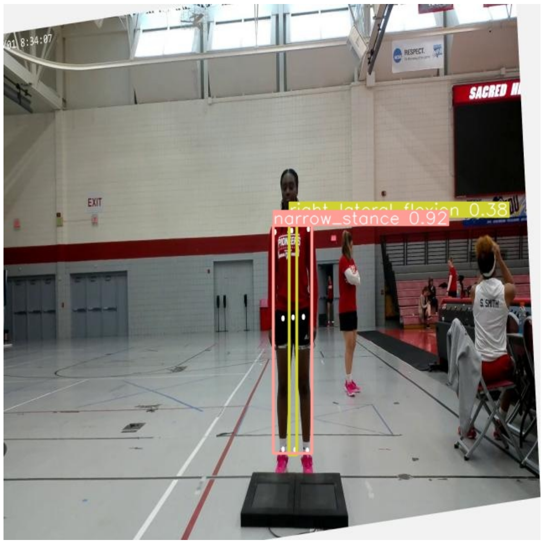
We used a video of 35 seconds for the evaluation of a single athlete, we extracted the frames at a rate of 6.028 frames per second to get 211 frames from the 35-second video. All of these frames were then fed into the MediaPipe Pose highlighting the necessary key points. To calculate the scores for the first time, we used MediaPipe Pose, however, the end goal was to eliminate the frame and key point extraction process.

B. Key Point Extraction

As mentioned above, we used MediaPipe Pose to get the essential key points from the body to calculate the lateral trunk flexion and the stance width. We did a comparative and feasibility analysis on different human pose estimation frameworks by reviewing [6].

TABLE I: Frame Characteristics

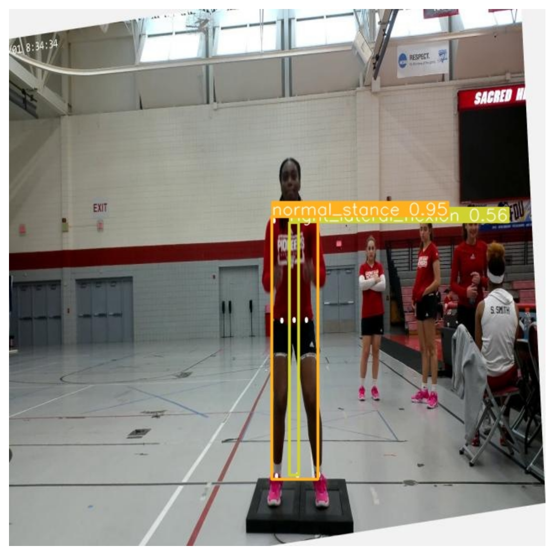
Frame Number	Lateral Flexion Present?	Lateral Flexion Right/Left/None	Stance Angle	Type (Normal/Wide/Narrow)	Ratio
7	Yes	Right	-1.636577042	Narrow	1.380668633
33	Yes	Right	-1.065845028	Normal	1.02102877
82	Yes	Right	-1.694647069	Normal	1.252205076
170	Yes	Right	-1.424974273	Normal	0.8873409484



(a) Frame 7

(b) Frame 33

(c) Frame 82



(d) Frame 170

Fig. 1: Predictions on Test Dataset

C. Calculation and Annotation

We annotate the frames with the key points after calculating the lateral flexion and stance width. We draw a straight line from the hip midpoint to the top of the frame and we draw another straight line from the hip midpoint to the shoulder midpoint. The angle between these lines is the lateral trunk flexion. If the angle isn't 0 or 180, the lateral trunk flexion error is present. Similarly, for Stance width, we calculate the Euclidean distance between the right ankle and left ankle, and the Euclidean distance between the right shoulder and left shoulder. The ratio of shoulder distance to ankle width is the stance width of the athlete. If the ratio is between 0.8 to 1.2 then it is the normal stance, whereas a ratio greater than 1.2 is a narrow stance and less than 0.8 is a wide stance.

We used roboflow [9] for annotating the frames and for creating the bounding boxes, these images will be later used as ground truth for a deep learning model to learn the presence of lateral trunk flexion and the values of stance width.

D. Model Selection

Upon annotating each individual frame of the video dataset with appropriate bounding boxes and their respective labels, the next step was to select a computer vision model which would train on that dataset with the manually added bounding boxes acting as the ground truth bounding boxes. Since our objective was to detect two errors in a given frame, we treated it as an instance segmentation task. Instance segmentation basically detects, segments and classifies every individual object in the image. Instance segmentation is a combination of semantic segmentation and object detection. Object detection localizes multiple objects with bounding boxes.

A semantic segmentation framework generates pixel-level category labels for each category class. Instance segmentation produces a segment map of each category as well as each instance of a particular class—therefore, providing a more meaningful inference on an image. [13] In this scenario, Instance segmentation would treat the errors present in the frame as objects and render two output bounding boxes for the respective errors. For instance segmentation, we conducted a comparison of two benchmark models to find the best suited model for our problem. The comparison can be found in the table III from [11] in the Discussions section.

Based on those discussions and consideration according to the requirement, the team decided to move forward with the YOLOv8 for the instance segmentation task. The model architecture comprises 168 layers, boasting an extensive parameter count exceeding 111 million. With a computational performance reaching 28.4 GFLOPs (billion floating-point operations per second), the model demonstrates a remarkable capacity for complex calculations. Notably, the processing speed is segmented into distinct stages: preprocessing (5.7ms), inference (20.3ms), and postprocessing (24.5ms per image). Additionally, during validation, no loss computation was performed, indicated by a negligible loss calculation time of 0.0ms. In the following section, we discuss how we trained our model.

E. Training

The YOLOv8 model pre-trained on the MS COCO dataset [12], was chosen for the instance segmentation task on our athlete dataset. As discussed in the 'Annotation' section, we provided the ground truth bounding boxes for the two errors namely Lateral Flexion and Stance Width to the pre-trained YOLOv8 model using Ultralytics [10] The dataset was exported from Roboflow to the Ultralytics, and the data was split in a 70:20:10 train : validate : test ratio respectively. Upon training, the model was able to provide bounding boxes for the errors.

Table II shows the performance of the model in predicting the errors as well as some characteristics of the data which inherently affected the prediction of the model. Since majority of the frames in the dataset consisted of right_lateral_flexion error and either normal_stance or narrow_stance, we can see that the model doing fairly well at predicting these errors whereas, it struggles to predict the left_lateral_flexion and wide_stance errors. This can be seen from the low recall and precision values. We presume that this is due to the fact that the dataset is skewed toward a certain error and this issue can be further solved by simply taking a larger inclusive dataset with every possible error with almost equal representation. Since there is a room for improvement as suggested by the moderate mAP values, a larger dataset could prove to be a breakthrough.

TABLE II: Model Performance on Validation Dataset

Class	Images	Instances	Box P	R	mAP@0.5	m
all	43	86	0.674	0.527	0.536	0.382
left_lateral_flexion	43	9	0.181	0.333	0.243	0.727
narrow_stance	43	22	0.884	**0.955**	**0.96**	0.873
no_lateral_flexion	43	1	0	0	0.6642	0.0204
normal_stance	43	19	0.579	**0.94**	**0.922**	0.779
right_lateral_flexion	43	31	0.403	**0.935**	**0.729**	0.366
wide_stance	43	2	0	0	0.3	0.183

III. Results

Table II shows the performance of the model on the validation dataset and the figure 1a,1b,1c, and 1a shows the prediction results on the test dataset which are matching with the ground truth boxes. As seen in table II the model precision is fairly high for certain errors than others. Regardless of the skewness, these results go on to show that computer vision approaches are ready for end to end services like live LESS scoring systems, error prediction systems, injury risk assessment systems, etc. We further discuss the implications of these results in the Conclusion and Future Work sections.

IV. Discussions

Since the goal of this project was to automate the Landing Error Scoring System (LESS) via jump landing videos and proposing a generalized framework that would work for any kind of dataset, it was crucial to pick a model suitable for our needs. Because we were treating this as an instance segmentation problem, the study conducted by R. Sapkota et al. [11] gave us a fair idea about the performance of the benchmarks. Table III from [11] summarizes the different performance metrics to compare YOLOv8 with Mask R-CNN, and it is clear from their results that for the segmentation tasks, YOLOv8 outperforms its two shot counter part. Apart from this, YOLOv8 offers faster inference compared to Mask R-CNN because it performs instance segmentation in a single pass, making it more suitable for real-time applications or scenarios where speed is crucial.

TABLE III: from [11] performance metrics of YOLOv8 and Mask R-CNN models for single and multi-class object segmentation tasks

Model	Precision	Recall	mAP@0.5	Inference Time (ms)	Frames Per Second (FPS)
YOLOv8 (Single-class)	92.9	97	0.902	7.8	128.21
Mask R-CNN (Single-class)	84.7	88	0.85	12.8	78.13
YOLOv8 (Multi-class)	90.6	95	0.74	10.9	91.74
Mask R-CNN (Multi-class)	81.3	83.7	0.700	15.6	64.10

V. Conclusion

Lower extremity injuries severely impact athletes' performance and careers. Therefore, coaches use neuromuscular monitoring and directed training initiatives to prevent these injuries. [1] The suggested framework aids coaches in identifying dangerous motions and improper postures. It strikes a compromise between the time-consuming and costly 3D motion analysis method and the clinician-driven screening that is hampered by intra- and inter-rater variability. Without depth, cameras, or skilled doctors, automation of the LESS using familiar 2D recordings would enable mass injury-risk screening programmes with feedback in almost real-time. If this method is successfully implemented, it could automatically identify people at a high risk of injury through smartphone applications that use LESS and 2D video footage. This would increase the accessibility of injury-risk assessment methods to a broader audience than elite athletes and eliminate the need for a depth-sensor camera [7]. Thus, the proposed framework for error prediction without the overhead of calculating the key points is a step forward towards automation of sports analytics with time efficient real time systems.

VI. Future Work

We had one video of 35 seconds and we were able to extract 210 frames from it. We trained the model from those 210 frames and we were able to get decent accuracy. However, if we had more frames, then it would have aided in adequately training the model, and subsequently the model could have learned the underlying pattern more efficiently. Furthermore, we have used only two LESS parameters out of the total 17, lateral trunk flexion and stance width. In future, the aim is to do annotations for all the 17 LESS parameters and make the end-to-end process automated.

References

[1] S. Sharma, S. Divakaran, T. Kaya, C. Taber, and M. Raval, "A Framework for Biomechanical Analysis of Jump Landings for Injury Risk Assessment," in Proceedings of the 26th IEEE Pacific Rim International Symposium on Dependable Computing, Oct. 2023, pp. 52, doi: 10.1109/PRDC59308.2023.00052.

[2] Zhu, D., Zhang, H., Sun, Y., Qi, H. (2021). Injury risk prediction of aerobics athletes based on big data and computer vision. Scientific Programming, 2021, 1-10.

[3] Guo L., Wu Y., Li L. (2019). Review on the Hot Issues of Core Stability Research. J. Shangdong Sport Univ. 35 (03), 113–118. 10.14104/j.cnki.1006-2076.2019.03.018

[4] Nealon A, Cook J, Docking S. Assessment of trunk lateral flexion range of movement using a novel method in first class cricket players. J Athl Train. 2021 Mar 3;56(12):1355–61. doi: 10.4085/564-20.

[5] Hershkovitz, A., Karni, O. (2018). Borders of change: A holistic exploration of teaching in one-to-one computing programs. Computers Education, 125, 429-443.

[6] Chung, J.-L.; Ong, L.-Y.; Leow, M.-C. Comparative Analysis of Skeleton-Based Human Pose Estimation. Future Internet 2022, 14, 380. https://doi.org/10.3390/fi14120380

[7] Hébert-Losier, K.; Hanzlíková, I.; Zheng, C.; Streeter, L.; Mayo, M. The 'DEEP' Landing Error Scoring System. Appl. Sci. 2020, 10, 892. https://doi.org/10.3390/app10030892

[8] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," [Online]. Available: https://github.com/facebookresearch/detectron2, 2019.

[9] Dwyer, B., Nelson, J., Hansen, T., et. al. (2024). Roboflow (Version 1.0) [Software]

[10] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLO," version 8.0.0, Jan. 2023. [Online]. Available: https://github.com/ultralytics/ultralytics.

[11] R. Sapkota, D. Ahmed, and M. Karkee, "Comparing YOLOv8 and Mask RCNN for object segmentation in complex orchard environments," 2023. [Online]. Available: https://doi.org/10.32388/ZB9SB0.

[12] T.-Y. Lin et al., "Microsoft COCO: Common Objects in Context," arXiv:1405.0312 [cs.CV], 2015.

[13] Bandyopadhyay, H. (2022, February 22). Instance Segmentation Guide. Retrieved from https://www.v7labs.com/blog/instance-segmentation-guide