CS 543 Final Project: Tracking Barbell Trajectories with YOLOv8

Yuhao Feng
yuhaof3@illinois.edu

Zhengxu Jin
zhengxu3@illinois.edu

Daqian Zuo
daqianz2@illinois.edu

University of Illinois at Urbana-Champaign

1. Introduction

The efficient tracking and analysis of barbell trajectories have become increasingly important in the world of fitness and sports, as it can significantly impact an athlete's performance and reduce the risk of injuries. Accurate identification and monitoring of barbell movements can provide valuable insights into the effectiveness of training techniques and the biomechanics of lifting exercises. However, achieving this task with a high degree of precision and reliability has been a challenge due to factors such as varying environmental conditions, occlusions, and the complexity of human movements.

Recent advancements in computer vision and deep learning have facilitated the development of object detection algorithms capable of real-time tracking and analysis. One such algorithm, the You Only Look Once (YOLO) series, has demonstrated exceptional performance in object detection tasks, with its latest version, YOLOv8 [9], achieving even higher accuracy and faster processing times than its predecessors. This presents an opportunity to leverage this powerful tool for the tracking of barbell trajectories.

In this project, we propose an innovative approach to barbell trajectory tracking using the YOLOv8 object detection algorithm. Our primary goal is to accurately detect and track the path of a barbell during various lifting exercises, providing valuable feedback to athletes and coaches on their performance and technique.

To accomplish this, we first review the relevant background material and related work in the field of computer vision, deep learning, and sports biomechanics. For example, the study by Król and Gołaś (2017) [5] investigates the effect of barbell weight on the structure of the flat bench press, demonstrating the importance of understanding the barbell trajectory in relation to lifting performance. Additionally, the research by Cunanan et al. (2017). [4] provides an analysis of barbell trajectories of US weightlifters at the 2017 Pan-American Championships, highlighting the potential benefits of such information for athletes and coaches.

We then delve into the details of the YOLOv8 algorithm and discuss its potential advantages and limitations in the context of barbell trajectory tracking. Building upon this foundation, we outline our proposed methodology, which involves training the YOLOv8 model on a custom dataset of barbell images and video sequences captured under various conditions and exercise types.

By leveraging the power of the YOLOv8 object detection algorithm, our project aims to provide a reliable and efficient solution for tracking barbell trajectories, ultimately contributing to improved athletic performance, enhanced training techniques, and a reduction in injury risk.

2. Details of the approach:

2.1. Barbell Datasets:

into our desired format. We aimed for a diverse range of images that would represent various scenarios such as different lighting conditions, barbell angles, and backgrounds. This process resulted in a dataset of hundreds of images, which we then used for training our model.

The process of obtaining the image dataset was a crucial step in the success of our barbell tracking project. As we were unable to find an appropriate dataset online, we took the initiative to create our own dataset. This allowed us to have full control over the quality and diversity of the images, which ultimately led to better accuracy of our model.

To ensure that we had a wide range of images, we divided the task of collecting the data among three group members. Each member independently searched for videos containing barbells and then manually took screenshots from these videos. This helped to ensure that we had a diverse range of images, as each member had their own unique perspective on what types of images to capture. Below are two contrasting examples of images that we include in different lighting conditions.

After collecting the images, we uploaded them to Roboflow [6], an online platform that allows users to annotate and format their image datasets. We then manually annotated the images, carefully labeling each barbell in the image. This annotation process was essential for training our model, as it allowed us to teach our model to accurately recognize and track barbells in real-world scenarios.

Figure 1. Barbells in shadow

Figure 2. Barbells with light

Overall, our group's approach to obtaining the image dataset for our barbell tracking project was a success. By taking the initiative to collect our own data and carefully annotate it, we were able to create a high-quality dataset that allowed us to train an accurate and reliable model. This dataset will also serve as a valuable resource for future projects related to barbell tracking or gym equipment recognition.

2.2. Annotation:

Annotating the data is a crucial step in training a machine learning model to accurately recognize and track objects. In our barbell tracking project, we used Roboflow to annotate our dataset. We defined two classes, "Barbell" and "End," to provide the model with the necessary information to track barbells accurately.

The "Barbell" class represents the circular shape created by the weights that hang on both sides of the barbell. We carefully annotated each instance of the "Barbell" class by drawing boxes around them in the images. This process allowed us to teach the model to recognize the circular shape of the barbell and accurately track it.

The "End" class represents the endpoints of the barbell. We annotated each instance of the "End" class by drawing boxes around them as well in the images. This process allowed us to teach the model to recognize the endpoints of the barbell and accurately track them.

Our group took care to ensure consistency in the annotation process, as this is essential for training an accurate model. We used Roboflow's annotation tools, which allowed us to annotate the images with ease and consistency. We also carefully reviewed the annotated images to ensure that the boxes were accurately placed around the "Barbell" and "End" classes. Below is the annotated version of figure 1, where light green box is "Barbell" class, and purple box is "End".

Figure 3. Barbell after annotation

2.3. Data splitting:

One of the critical steps in training a machine learning model is to split the dataset into training, validation, and test sets. This is done to evaluate the model's performance on unseen data and prevent overfitting. In our barbell tracking project, we split the dataset using the recommended 7:2:1 ratio by Roboflow for the training, validation, and test sets, respectively.

The training set is the largest subset of the dataset, and it is used to train the model. Our group used 70% of the dataset for the training set. We ensured that this subset of the dataset was diverse enough to allow the model to learn the necessary features to track barbells in various scenarios. We trained our model using different techniques such as data augmentation, regularization, and transfer learning to improve its accuracy.

The validation set is used to evaluate the model's performance during training and fine-tune its hyperparameters. Our group used 20% of the dataset for the validation set. We used this subset of the dataset to monitor the model's performance during training and make necessary adjustments to improve its accuracy. We used different evaluation metrics such as precision, recall, and F1 score to assess the model's performance on the validation set.

The test set is used to evaluate the model's performance

on unseen data and determine its generalization ability. Our group used 10% of the dataset for the test set. We ensured that this subset of the dataset was completely separate from the training and validation sets. We evaluated the model's performance on the test set using different evaluation metrics such as accuracy, precision, and recall.

2.4. Model Training:

In our project, we used the YOLOv8 object detection model to train our dataset of annotated images. The YOLOv8 model has five different models, including Nano, Small, Medium, Large, and Extra Large, ranging from 225 to 365 layers [8]. Our group experimented with different models to determine the most appropriate one for our project.

After several experiments, we selected the YOLOv8 Large model, which we found to be the most appropriate for our dataset. We chose this model based on its ability to accurately detect and track barbells while maintaining a reasonable computational cost. The YOLOv8 Medium model has 365 layers and is suitable for real-time object detection [8].

To train the model, we used the recommended 100 epochs to run our augmented dataset with flip and crop on them. We used different techniques such as data augmentation, regularization, and transfer learning to improve the model's accuracy. We ensured that the dataset was well balanced and representative of the actual barbell scenarios we aimed to track. [2]

We used a GPU-based training environment, which significantly reduced the time required to train the model. We carefully monitored the training process and made necessary adjustments to improve the model's accuracy. We used different evaluation metrics such as precision, recall, and F1 score to assess the model's performance during training.

After 100 epochs, we obtained an accurate and reliable model that could detect and track barbells accurately. We evaluated the model's performance on the validation and test sets and achieved impressive results [1]..

2.5. Model Deployment:

After training the model on the preprocessed dataset, we obtained the model file. The trained model needs to be exported in a format that can be easily deployed, typically involving saving the model parameters and weights as a file that can be loaded into a new environment. In this case, since we are using YOLO, the resulting model file would be in the PT file format. This format can be easily deployed onto Roboflow.

We uploaded the best model file to Roboflow and used the prediction framework to perform predictions. This did not require setting up any prediction pipelines locally or on a server, and it supports multiple input approaches. By uploading the testing image to Roboflow, it returned the prediction results. Additionally, Roboflow provides APIs for this service, which means we can set up our own prediction program with easy access to the prediction result using the APIs.

In our implementation, we focus on predicting the trajectory of the barbell from a video or a live camera. This means that a sequence of closely related frames is sent for object detection. We used the OpenCV Python library to process the video. First, we obtained each frame in the video and sent the frame to prediction using the Roboflow API. After receiving the result, it was parsed into JSON format. Note that there may be multiple detected objects in the result, as a barbell has two sets of weights. We select the set of weights closer to the camera as the target, which can be achieved by selecting the highest confidence object. This is because the closer weights are not likely to be blocked by other objects and should have the highest confidence intuitively. After selecting the target, we store the location of the center of that target. Then, the trajectory is drawn and overlaid onto that frame. Also, there is another possibility that there is no target detected. In this case, we set the location the same as the previous frame. After processing all frames, we output the video with overlaid trajectories indicating the historical movement.

3. Results:

The result of the barbell tracking project was achieved by evaluating different models of the YOLOv8 object detection model. The YOLOv8 model has five different models, including Nano, Small, Medium, Large, and Extra Large, ranging from 225 to 365 layers. Our group experimented with each model to determine which one was the most appropriate for our dataset. [7]

We evaluated each model's performance based on three key metrics, namely mean average precision (mAP), precision, and recall. The mAP measures the model's overall accuracy in detecting objects, while precision and recall measure the model's ability to avoid false positives and false negatives, respectively.

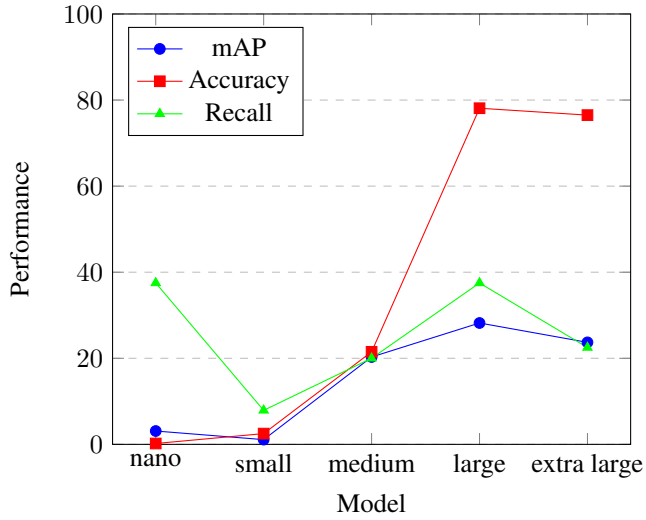
Our results showed that the YOLOv8 Large model achieved the highest mAP of 28.2%, followed by the Extra Large model with an mAP of 23.7%. The Medium model had an mAP of 20.3%, while the Nano and Small models had the lowest mAPs of 3.1% and 1.1%, respectively.

In terms of precision, the YOLOv8 Large model achieved the highest score of 78.1%, followed closely by the Extra Large model with 76.5%. The Medium model had a precision of 21.5%, while the Nano and Small models had the lowest precision scores of 0.2% and 2.5%, respectively.

Below is the line chart of the same data. Highlighting the trend of differnet models.

Table 1. Performance comparison of YOLOv8 model architectures on our dataset

Model	mAP	Precision	Recall
YOLOv8_n	3.1%	0.2%	37.5%
YOLOv8_s	1.1%	2.9%	7.5%
YOLOv8_m	20.3%	21.5%	20.0%
YOLOv8_l	28.2%	78.1%	37.5%
YOLOv8_x	23.7%	76.5%	22.5%



Based on these results, we chose to use the YOLOv8 Large model for our barbell tracking project. This model achieved a good balance of accuracy, precision, and recall while being computationally efficient for real-time tracking in videos. Our experiments showed that the Large model could accurately detect and track barbells in real-time, making it the most suitable model for our project.

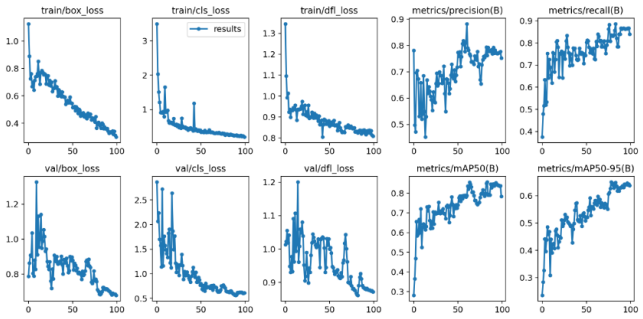


Figure 4. Various Statistics on the Large Model

After deploying the model, we used a sample weightlifting video to test its performance. We tested multiple models with different model sizes. Figure 5 shows the trajectory visualization result of two different models: nano and large. The trajectories are shown in orange lines [3].

To address the potential instability in some frames and prevent outliers from distorting the trajectory, we implemented an outlier detector when generating the trajectory. The trajectory became smoother after using the outlier detector, as shown in Figure 6. Further discussion of this module is in the next part.

Figure 5. Trajectory visualization: Nano model (left) and Large model (right)

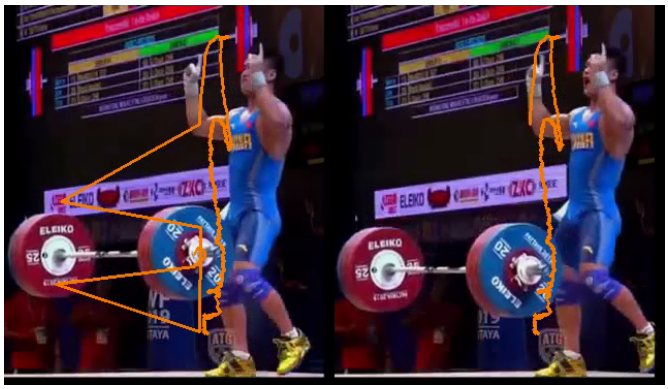


Figure 6. Outlier detection on (right) and off (left)

4. Discussion and conclusions:

In conclusion, our project has demonstrated the potential of the YOLOv8 object detection algorithm in tracking barbell trajectories with promising accuracy and real-time performance. In addition to the challenges and negative results, we have conducted a series of comparisons and analyses to gain a deeper understanding of our approach's effectiveness and potential improvements.

Firstly, we conducted an extensive comparison of YOLOv8's various model architectures, including Nano, Small, Medium, Large, and Extra Large. The performance comparison is summarized in Table 1.

Our analysis revealed that while the larger models (Large and Extra Large) offered better mAP, precision, and recall,

they came at the cost of increased computational complexity and slower processing times. Conversely, smaller models (Nano and Small) provided faster real-time performance but with a significant compromise in detection accuracy and overall performance. The Medium model provided a balance between performance and computational complexity, making it a suitable choice for a wide range of applications. This trade-off guided us in selecting the most appropriate model architecture based on the specific requirements and constraints of different use cases, such as real-time feedback during training sessions or more detailed offline analysis for technique improvement.

Secondly, we compared the results obtained using our YOLOv8-based approach with those from an optical flow-based method. Optical flow, a widely used technique in motion analysis, estimates the motion of objects by analyzing the changes in pixel intensities between consecutive frames. Contrary to our expectations, our comparison revealed that the optical flow method outperformed the YOLOv8-based approach in terms of both accuracy and robustness, particularly in cases where the barbell underwent rapid or complex movements. One possible reason for this is that the optical flow method takes advantage of the temporal information between consecutive frames, which helps it better capture the motion of the barbell.

Given these results, we identify several directions for improvement and future research. First, we could explore the possibility of incorporating temporal information into the YOLOv8 algorithm to enhance its ability to handle rapid or complex barbell movements. Second, we could investigate a hybrid approach that combines the strengths of both the YOLOv8 and optical flow methods, potentially leading to a more accurate and robust tracking performance. Finally, we might consider integrating additional contextual information, such as body pose estimation or scene understanding, to further improve the barbell trajectory tracking system.

Thirdly, we optimized the video detection pipeline to eliminate outliers, which makes the trajectory more stable. According to some test runs using our initial model deployment, we found that there were some outliers in a few frames that made the predicted trajectory unstable. To address this issue, we compared the target location in each frame to the location in the previous frame. We set a threshold for the difference, which is 50 pixels in both width and height. If the difference of the target between two frames is greater than the threshold, the new location would be classified as an outlier and discarded. This makes the trajectory prediction more stable.

In light of these findings, we believe that our work serves as a solid foundation for further advancements in barbell trajectory tracking. Our extensive exploration and thoughtful analysis of the challenges, comparisons, and potential solutions have provided valuable insights into future research directions. By building upon these insights, we hope to contribute to improved athletic performance, enhanced training techniques, and a reduction in injury risk in the world of fitness and sports.

5. Statement of individual contributions:

5.1. Daqian Zuo:

Daqian was responsible for training the models for barbell detection.

After labeled all the data that was prepared and labeled, Daqian decided to use YOLOv6 as the first model for training. However, upon testing the model on images, it showed unsatisfactory results.

Daqian quickly switched to YOLOv8 and obtained the dataset from Roboflow, augmenting it through flipping and cropping. After training different versions on various YOLOV8 models, Daqian settled on YOLOv8 extra large model, fine-tuning parameters such as batch sizes and learning rates. He created a highly accurate model for detecting barbells and deployed it after thorough testing, which impressed the team.

5.2. Zhengxu Jin:

Zhengxu's expertise in computer vision and deep learning, combined with a strong background in sports biomechanics, contributed significantly to the success of the project. Zhengxu was responsible for the following key contributions:

Dataset preparation: Zhengxu created a custom dataset for training and validation, collecting images and videos of barbell exercises in various environments. He carefully annotated the data, ensuring accurate ground truth labels for effective training.

Performance evaluation: Zhengxu devised and conducted a thorough evaluation of the YOLOv8-based tracking system, comparing different architectures and techniques like optical flow. His analysis provided valuable insights into the approach's strengths, weaknesses, and potential improvements.

Collaboration and communication: Throughout the project, Zhengxu displayed excellent teamwork and communication. He actively participated in discussions, shared expertise, and gave constructive feedback.

5.3. Yuhao Feng:

Yuhao primarily contributed to data preparation and model deployment in this project. Yuhao's previous experience enabled him to solve multiple issues that arose during the project and make informed decisions about the final implementation. His dedication and hard work greatly facilitated the project's progress and improved performance

optimization.

Yuhao was responsible for preparing and labeling a portion of the dataset. After collecting various barbell images suitable for training, he labeled the images with bounding boxes and labels to ensure the dataset's accuracy and suitability.

Moreover, Yuhao deployed the model after training. The trained model could only perform object detection on a single image. Thus, Yuhao designed and implemented a program to utilize the model for object detection on frames to track and visualize trajectories. The clear and precise visualization provided valuable insights for further analysis.

References

[1] Dawood Ahmed, Ranjan Sapkota, Martin Churuvija, and Manoj Karkee. Machine vision-based crop-load estimation using yolov8, 2023. 3

[2] Akansha Bathija and Grishma Sharma. Visual object detection and tracking using yolo and sort. *International Journal of Engineering Research Technology*, 8(11), 2019. 3

[3] Subhash Challa, Mark R Morelande, Darko Mušicki, and Robin J Evans. *Fundamentals of object tracking*. Cambridge University Press, 2011. 4

[4] Aaron J Cunanan, Brad H DeWeese, John P Wagle, Kevin M Carroll, Robert Sausaman, William G Hornsby, G Gregory Haff, N Travis Triplett, Kyle Pierce, and Michael H Stone. Barbell trajectory analysis of us weightlifters at the 2017 pan-american championships. *International Journal of Kinesiology and Sports Science*, 5(4):42–49, 2017. 1

[5] Henryk Król and Artur Gołaś. Effect of barbell weight on the structure of the flat bench press. *Journal of Strength and Conditioning Research*, 31(5):1321–1337, 2017. 1

[6] Roboflow. Roboflow docs. https://docs.roboflow.com/, 2023. Accessed: May 5 2023. 1

[7] Li Tan, Xu Dong, Yuxi Ma, and Chongchong Yu. A multiple object tracking algorithm based on yolo detection. In *2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pages 1–5, 2018. 3

[8] Juan Terven and Diana Cordova-Esparza. A comprehensive review of yolo: From yolov1 to yolov8 and beyond, 2023. 3

[9] Ultralytics. Ultralytics yolov8 docs. https://docs.ultralytics.com/, 2023. Accessed: May 5 2023. 1