Report for In-Video Trajectory Overlay for cricket ball identification and tracking

Modelling Decisions:

For this project, YOLO-v11 was chosen for finetuning to detect the cricket ball within the static camera match footage. YOLO was selected among multiple deep learning detection models for its strong real-time object detection and robust handling of fast-moving, small objects, which is a strict requirement given the size and speed of the cricket ball. The other reason is the ease of dataset creation and model fine-tuning for YOLO. There are several open-source packages readily available to aid in annotating and generating the dataset required for training this model. By training the model on the originally annotated and augmented match frames, it can handle variations in ball appearance, lightning, and different camera angles. This type of training creates a more reliable model than the base model alone. The following are some modelling decisions that were made:

- Data Augmentation techniques such as brightness change, random zoom, horizontal flip, Gaussian blur, and scaling were used for augmenting the annotated frames.
- Only a single target class, “cricket_ball”, helped the model learn robustly and reduce false detections on other small objects.
- Only a few hyperparameters (image size, epochs, batch size) were altered to get different results because of the time constraints.

Fallback Logic:

To improve robustness, fallback logic was used, which allowed for handling frames where the ball was not visible properly.

- **Confidence thresholding:** The confidence score was kept high (0.5) while generating predictions to reduce the false positives.
- **Trajectory interpolation**: After the calculation of the centroid, any frames that skipped the prediction of the ball, even when it is present in the frame, had an interpolated trajectory so that the trajectory line remained continuous.

Assumptions Made:

Several assumptions were made during the data generation and model training.

- Only one cricket ball is present throughout one video.
- Training data sufficiently captures the variations in ball sizes, color, and lighting.
- No other object of comparable size, speed, or color is present in the videos.

Fixes done to make the model more robust:

There were several errors that were faced and had to be fixed.

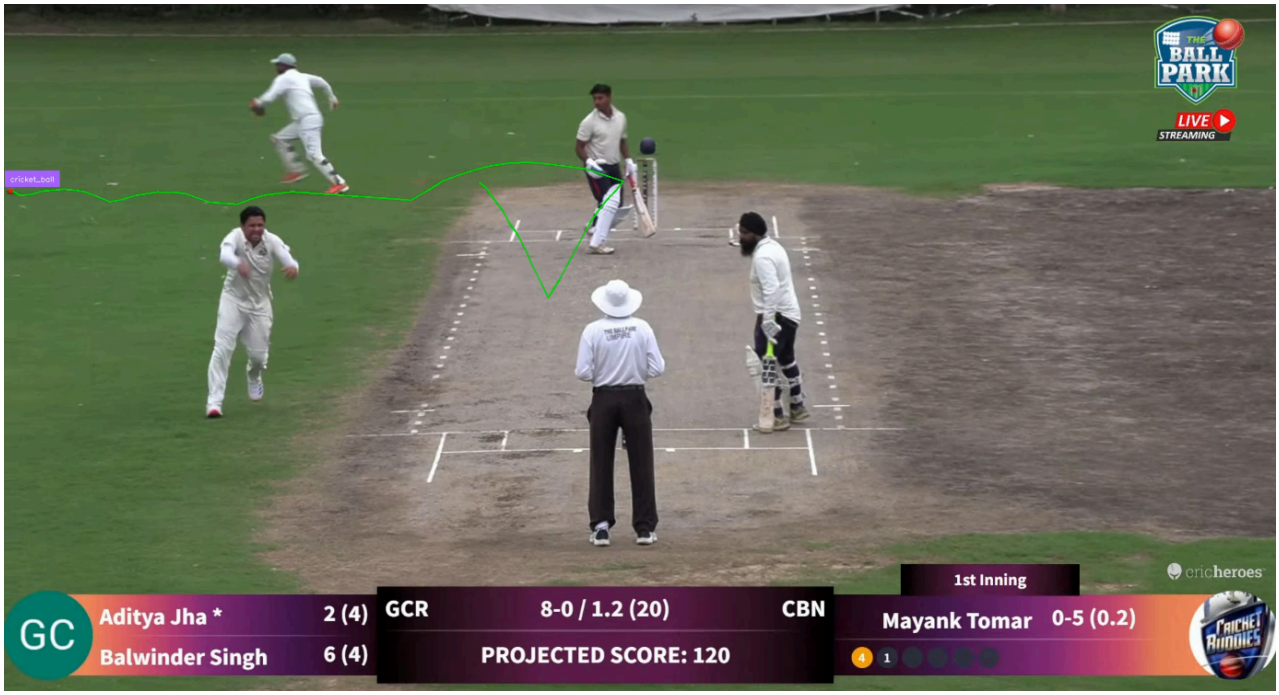
- The model, after training for 100 epochs, showed consistent misclassifications of white cricket balls on the shiny helmets of the players. This was omitted by increasing the number of epochs to 150.
- When the confidence score was set too low, the model misclassified the players' white shoes as the cricket ball. The confidence score was then increased to 0.5 to omit this error.
- The model, after training, still missed a few frames for ball detection, making the trajectory discontinuous. This error was reduced by interpolating the trajectory through the last detected frame and the current one. This improved the result significantly.

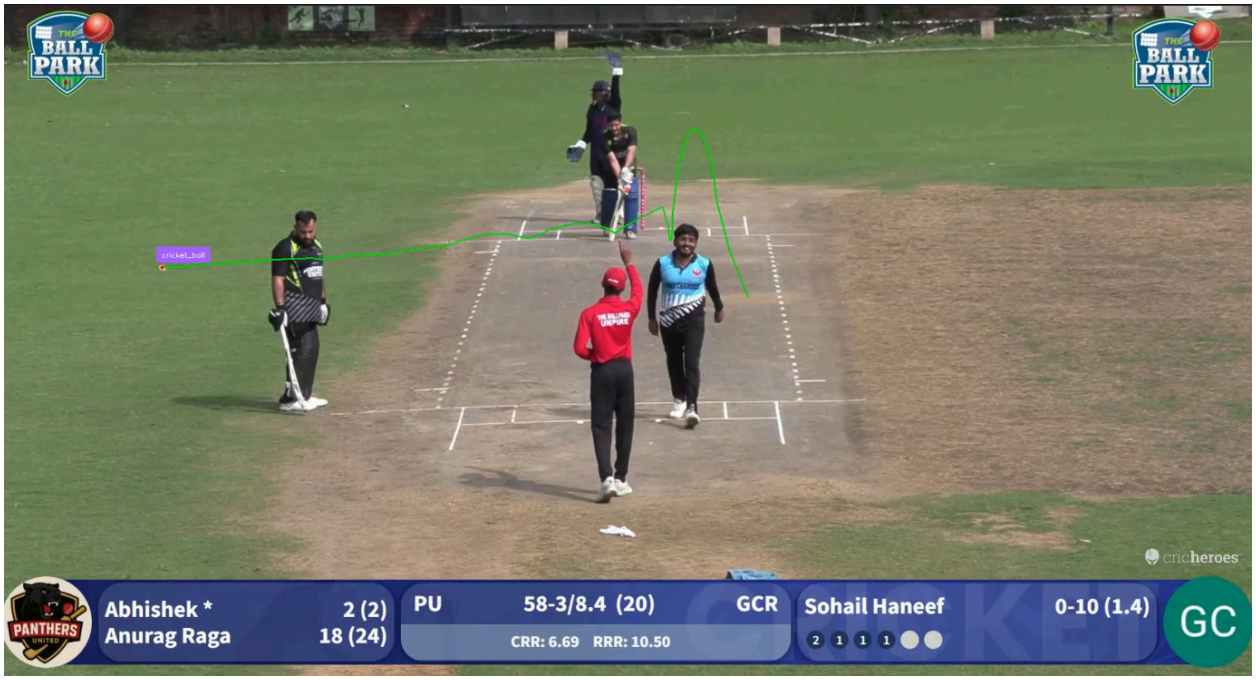
Train/Test data and Limitations

The first 8 videos (1.mp4 to 8.mov) were annotated manually using the annotation software, which took a significant amount of time. The rest of the videos were used to evaluate the effectiveness of the trained model.
Since the type of videos (lighting conditions, texture, and angle of the camera) that were annotated differed from some test videos, the model didn't perform as well on the videos whose types were not used for training.

Example Outputs

Below are some examples of test video frames that were received after trajectory identification.




THE BALL PARK
cricket_ball
cricheroes
PANTHERS UNITED
Abhishek * 2 (2)
Anurag Raga 18 (24)
PU 58-3/8.4 (20) GCR
CRR: 6.69 RRR: 10.50
Sohail Haneef 0-10 (1.4)
GC


AirDrop
Everyone
Do Not Disturb On
Sound
Display
Edit Controls
cricheroes
Mavericks 100-1 10.1
CRR: 9.84 PROJECTED SCORE: 197
vs The Rising Squad
S. VIJAY REDDY 52(34)
PAVAN CHERUKURI 15(9)
RAKESH M 0-2(0.1)
OV.

An instance where the model lacks in identifying the ball and the trajectory becomes incomplete.