**Initial Approach and Challenges**

**CNN model for object classification:**

At the outset, the task of detecting the four corners of a cricket bat appeared relatively easy. Initially, I tried developing a Convolutional Neural Network (CNN) model using transfer learning with VGG16 as the base architecture and use Harris Corner Detection algorithm in OpenCV to detect the corners. While the CNN model gave an average accuracy, it soon became evident that it could only classify one of the corners present in the image and the corner detection algorithm couldn’t detect anywhere near the bat. This was my initial challenge.

**Exploration of Custom Object Detection:**

Recognizing the need for precise corner detection rather than classification, I focussed on an exploration of custom object detection solutions. This involved extensive research, encompassing various online resources, articles, and forums.

**Initial Attempts with TensorFlow API:**

One of the initial attempts involved using TensorFlow's object detection API. However, this approach did not yield the expected results.

**Transition to YOLOv8 as the Base Model:**

Subsequently, I tried with several YOLO versions, and YOLOv8 offered the most accurate results for our specific cricket bat corner detection task.

**Model Development**

**Data Collection:**

Data collection was time-consuming but crucial step. I gathered images from various sources, including sports websites and online shopping sites. The objective was to gather a diverse and representative dataset of cricket bat images, including various brands, styles, and angles.

**Data Annotation:**

Following data collection, each corner of cricket bat in the image was labelled with four classes - ‘top left’, ‘top right’, ’bottom left’, ’bottom right’. LabelImg software was used for the same. This was the most time-consuming aspect in the project as the accuracy of annotations was essential for training and creating a good model.

**Data Splitting:**

To ensure proper model training, the annotated dataset was divided into three distinct sets: training, validation, and test. Each set have two subfolders – images (with the images), labels (with the annotations in yolo format)

**Model Creation:**

The model was created in a google colab notebook with YOLOv8 as the base architecture. Specific code sets for installation and transferring weights was sourced from ‘Ultralytics’, developer of YOLOv8. The main link between the Yolo model and our custom dataset is the ‘data.yaml’ file, which has the path to our datasets and the class labels in the images. I transferred the YOLOv8 weights into our dataset.

Initially the number of images where less and the accuracy was not up to the mark. Then I added more diverse images , increased the number of epochs and changed image size which resulted in a decent mAP (Mean Average Precision) above 70%. The best and last weights of our model were automatically saved in the directory as a pt file (Pytorch model). The best weights were taken for our model and was saved as ‘bat\_corner\_model.pt’

**Final Test and Output:**

The sole objective of the whole project was to detect the four corners of the bat. The given input image was tested with the created model and it was detecting the corners accurately as shown below.

Output:



I also created a separate python file to load the developed model and test it with several images. The four corners of the bat were accurately detected for different images. Then the given image was tested and it was able to detect the corners as shown below.

Output:



The accuracy could be further improved by expanding the dataset with more images and adjusting the parameters. However, due to time constraints, I opted to submit the project upon achieving the desired objectives.

Completing this project has been an invaluable learning experience, and I'm grateful for giving me an opportunity to grow and learn.

Sincerely,

Rocky Joson

GitHub Repo Link: