```
        t2 = time.time()

        pred_bbox = [tf.reshape(x, (-1, tf.shape(x)[-1])) for x in pred_bbox]
        pred_bbox = tf.concat(pred_bbox, axis=0)

        bboxes = postprocess_boxes(
            pred_bbox, original_frame, input_size, score_threshold)
        bboxes = nms(bboxes, iou_threshold, method='nms')

        # extract bboxes to boxes (x, y, width, height), scores and names
        boxes, scores, names = [], [], []
        for bbox in bboxes:
            if len(Track_only) != 0 and NUM_CLASS[int(bbox[5])] in Track_only
or len(Track_only) == 0:
                boxes.append([bbox[0].astype(int), bbox[1].astype(int),
bbox[2].astype(
                    int)-bbox[0].astype(int), bbox[3].astype(int)-
bbox[1].astype(int)])
                scores.append(bbox[4])
                names.append(NUM_CLASS[int(bbox[5])])

        # Obtain all the detections for the given frame.
        boxes = np.array(boxes)
        names = np.array(names)
        scores = np.array(scores)
        features = np.array(encoder(original_frame, boxes))
        detections = [Detection(bbox, score, class_name, feature) for bbox,
                      score, class_name, feature in zip(boxes, scores, names,
features)]

        # Pass detections to the deepsort object and obtain the track information.
        tracker.predict()
        tracker.update(detections)

        # Obtain info from the tracks
        tracked_bboxes = []
        for track in tracker.tracks:
            if not track.is_confirmed() or track.time_since_update > 5:
                continue
            bbox = track.to_tlbr()  # Get the corrected/predicted bounding box
            class_name = track.get_class()  # Get the class name of particular
object
            tracking_id = track.track_id  # Get the ID for the particular track
            # Get predicted object index by object name
```