Exercise 20 & 21 - Object Detection & Tracking

Đặng Linh Anh

March 2021

1. (Optional) Các bạn tìm hiểu về Intersection over Union metric (IoU) và áp dụng IoU để loại bỏ các bounding box trùng lặp trong bài toán object detection.

Lời giải

Intersection over Union (IoU, hay còn được gọi là chỉ số Jaccard) là chỉ số đánh giá một mô hình dự đoán (bao gồm: object detection, segmentation, v.v.) trực quan và phổ biến nhất hiện tại.

IoU được tính toán như sau:

$$\mathbf{J}(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Chỉ số này có giá trị thuộc $[0, 1]$ và tăng dần khi kết quả dự đoán tiến gần và trùng với giá trị ground-truth.

IoU là metric đơn giản đủ để đánh giá đồng thời tọa độ và diện tích của kết quả dự đoán so với giá trị thực.

Trong bài toán object detection, để mạng deep CNN hoạt động được, trước tiên cần phải thực hiện map các default box (tương đương với region proposals trong họ 1R-CNN và anchor box trong YOLO hay SSD) với các ground-truth bounding box. Việc tính toán này dựa trên chỉ số IoU của các default box so với từng grond-truth box để phân chia thành positive và negative. Vì thao tác này thực hiện trên một giá trị IoU ngưỡng nên trên mỗi ground-truth box nhiều khả năng được map với nhiều default box. Mô hình sẽ được huấn luyện đồng thời trên nhiều default box như vậy. Đây cũng là nguyên nhân sau khi đi qua mô hình, kết quả sẽ xuất hiện sự trùng lặp các bounding box với nhau (cùng chỉ vào một đối tượng). Để xuất ra bounding box tối ưu, chính xác nhất ứng với object đó, ta dùng kỹ thuật Non-maximum Suppression (NMS).

Non-maximum Suppression xem xét trên mỗi ground-truth box, giữ lại những kết quả có IoU so với ground-truth box cao hơn giá trị ngưỡng cho trước. Sau đó, giữ lại kết quả có chỉ số confidence cao nhất trong số các boxes thu được.

Cụ thể giải thuật NMS được diễn giải qua hình dưới đây:

Algorithm 1 Non-Max Suppression

procedure NMS(B,c)
  $B_{nms} \leftarrow \emptyset$ Initialize empty set
  for $b_i \in B$ **do** => Iterate over all the boxes
    $discard \leftarrow$ False Take boolean variable and set it as false. This variable indicates whether b(i) should be kept or discarded
    for $b_j \in B$ **do** Start another loop to compare with b(i)
      if same$(b_i, b_j) > \boldsymbol{\lambda_{nms}}$ **then** If both boxes having same IOU
        if score$(c, b_j) >$ score(c, b_i) **then**
          $discard \leftarrow$ True Compare the scores. If score of b(i) is less than that of b(j), b(i) should be discarded, so set the flag to True.
    if not $discard$ **then** Once b(i) is compared with all other boxes and still the discarded flag is False, then b(i) should be considered. So add it to the final list.
      $B_{nms} \leftarrow B_{nms} \cup b_i$
  return B_{nms} Do the same procedure for remaining boxes and return the final list

Hình 1: Giải thuật Non-maximum Suppression

2. (Optional) Từ source code được cung cấp cho bài toán object tracking, các bạn thiết kế lại thuật toán để giải quyết vấn đề object có thể xuất hiện với kích cỡ khác nhau.

Lời giải

Vấn đề object xuất hiện với kích cỡ khác nhau trong bài toán object detection/tracking được đề cập từ khá sớm trong những mô hình sơ khai như OverFeat, R-CNN với tên gọi "Scale Invariance". Những mô hình sau đó được cải tiến và có những nghiên cứu nghiêm túc về vấn đề này, có thể được kể đến như: SPPNets của Kaiming He - sử dụng Image Pyramid để detect các vật thể, Fast R-CNN - đề cập phương pháp Brute Force Learning với việc sử dụng ảnh input đầu vào được scale một cách ngẫu nhiên trên ngưỡng cho phép.

Trong bài toán Object Tracking lần này, để phù hợp với tính đơn giản của thuật toán, em đề xuất thuật toán như sau:

i. Xác định các giá trị scale mặc định (các giá trị scale này cần đối xứng qua điểm 1, ví dụ: $\{0.8, 0.9, 1.1, 1.2\}$)

ii. Trích xuất thủ công template là bounding box cho object trên frame đầu tiên, ta có được các giá trị top-left và 2 chiều kích thước của template.

iii. Tại frame kế tiếp:

- Thực hiện Sliding Search với window cùng kích thước với template để tìm ra template có cosine similarity cao nhất so với template tìm được ở frame trước. Tính toán tuyến tính, ta tìm được kích thước bounding box.
- Ứng với mỗi giá trị scale, thực hiện scale bounding box tìm được và feed vào mạng neural để tính template feature mới. Resize bounding box ở frame trước với cùng scale và feed vào mạng. So sánh 2 feature map này ta được điểm cosine smilarity mới. Giá trị scale nào cho ra score cao hơn giá trị đã tính ở trên, dùng template mới ứng với giá trị này để match cho frame tiếp theo.

iv. Lặp lại bước (ii).

Thuật toán được thực nghiệm trong NOTEBOOK.