百川學士學位學程大一下專題探索

利用 Detr 建立辨識 3D 物體模型

孫奇霆

摘要

本專題將實作一個 DETR 3D 物體辨識模型，並利用 coco 數據集進行測試。修改超參數後比較 DETR 在不同圖片解析度下的準確度。並將測試結果與模型可視化，探討 DETR 內部運作原理。

動機

先前的物體檢測模型，以 RCNN 為主，缺點是並行性較差。而 Facebook AI 於 2020 年提出的 DETR（Detection Transformer）則是一種基於 Transformer 架構的物體檢測模型，它利用了 Transformer 的自注意力機制來捕捉圖像中不同位置間的相互關係。據說不需要調整許多的超參數即可得到相當或超越 RCNN 的結果，這是否意味著模型的適應能力很強？本專題將探討調整剩下的超參數對結果的影響，來確定 DETR 究竟是具有良好的適應性，抑或是仍然需要調整至恰當的超參數才有這種效果。

文獻探討

最簡單的物體檢測流程是：(1)輸入圖像，用一個滑動的窗口擷取一小塊圖片；(2)將各區域傳遞給卷積神網絡[4](CNN)，並將區域分類；(3)一旦將每個區域劃分為相對應的類別後，就可以組合這些區域，來檢測原始圖像。(4)重複 1~3 直到每個位置、每種大小的窗口都嘗試完畢。

這種方法的優點是直接使用分類模型就可以做到物件辨識；但這種方法有兩大缺點：(1)他花費了大多數的時間在不必要的圖片上，因此如何將圖片有效率切分就是一項挑戰。(2)一個物體可能會被不同的窗口多次檢測到，如何融合這些數據又是一項挑戰。

為了解決窗口切分的效率問題，R-CNN[3](RNN+CNN)引入了區域提議(Region Proposals)，先找出有可能的候選位置，然後對候選區域進行分類。後來 Fast-RCNN[2]與 Faster RCNN 也是類似的作法，只是速度上加速非常多。但無論是哪一種方法，都沒有解決時間浪費在不必要之圖片的問題，且 RCNN 的並行性極差，導致訓練和推論效率低下。

隨著 Transformer[1]的出現，由於良好的並行性，很大程度上取代了 RNN 這

種遞迴機制，針對物體檢測的 DETR[6](Detection Transformer)也應運而生。

DETR 融合了 CNN 與 Transformer 的優勢，前者善於從像素點中分離出不同的局部特徵，而後者善於從大量的局部訊息中找到全局特徵。DETR 不需要區域提議，物體也不會同時被不同的窗口多次檢測到，一次解決傳統 RCNN 的兩大痛點。以下是 DETR[6]模型的一些主要特點和組件：

1. End-to-End：DETR 是一個端到端訓練的模型，它不依賴於傳統的區域提議網絡（RPN），因此和非極大值抑制（NMS）等手段，省下了調整許多參數的麻煩，因而簡化了物體檢測流程。

2. Transformer 架構：該架構最初被設計用於自然語言處理（NLP）任務，但其注意力機制在圖片辨識上同樣取得巨大成功。Transformer 幫助模型理解圖像中的上下文信息，並處理物體之間的關係。

3. Bipartite Matching Loss：DETR 引入了一種名為雙邊匹配損失（Bipartite Matching Loss）的新型損失函數，用於在訓練過程中匹配預測邊界框和真實邊界框。這種方法解決了物體檢測中的賦值問題。

4. 物體和背景的區分：由於 decoder 一次的輸出為固定長度，因此 DETR 使用一種稱為「無物體類別」（"no object" class）來區分圖像中的物體和背景，這有助於模型更準確地進行物體檢測。

5. 全局理解：通過自注意力機制，DETR 能夠一次全局地理解圖像，不像 CNN 需要一層一層傳遞，這使得模型較 CNN 能夠處理一些複雜的場景，尤其在圖片像素量很多時尤為明顯。

DETR 提供了一種簡單且統一的檢測架構。由於其較不需要依賴人類經驗且擁有優異的並行性能，DETR 已成為計算機視覺領域的重要研究對象。

目標

- 利用 PyTorch 實作 DETR 基本架構
- 微調超參數及改變架構
- 將 backbone 結果及 transformer 注意力機制可視化

研究工具

本專題會運用 COCO 數據集[5]，做為模型建立的測資。COCO 數據集是一個大型開源圖片數據集，可以用來作物件辨識圖像數據集。

COCO 數據集可以用於 CV 領域的各類研究，如：Detection, Segmentation, Keypoints……，有 33 萬張以上的影像（其中超過 20 萬張影像已標記），還包

含 150 萬個物件、並分成 80 個類別（for object detection）以及 91 類的 stuff（for semantic scene labeling），也正因為它的強大，所以現在 CV 領域的主流研究幾乎都會採用這份資料集來測試及驗證他們的方法。

COCO 數據集的官網同時也提供一組 API(一般都稱之為 COCO API)來協助加載，解析和可視化 COCO 中的標註(annotations)。API 支持對象實例(object instance)，對象關鍵點(object keypoints)和圖像標題(image caption)標註(annotation)資訊的提取。

研究方法

本專題的運作流程與方法如下：

1. 實現 DETR 模型

```
class DETR(nn.Module):
    def __init__(self, num_classes, hidden_dim, nheads,
                 num_encoder_layers, num_decoder_layers):
        super().__init__()
        self.backbone =
nn.Sequential(*list(resnet50(weights=ResNet50_Weights.DEFAULT).children())[:-2])
        self.conv = nn.Conv2d(2048, hidden_dim, 1)
        self.transformer = nn.Transformer(hidden_dim, nheads,
                                          num_encoder_layers, num_decoder_layers)
        self.linear_class = nn.Linear(hidden_dim, num_classes + 1)
        self.linear_bbox = nn.Linear(hidden_dim, 4)
        self.query_pos = nn.Parameter(torch.rand(100, hidden_dim))
        self.row_embed = nn.Parameter(torch.rand(50, hidden_dim // 2))
        self.col_embed = nn.Parameter(torch.rand(50, hidden_dim // 2))

    def forward(self, inputs):
        x = self.backbone(inputs)
        h = self.conv(x)
        H, W = h.shape[-2:]
        pos = torch.cat([
            self.col_embed[:W].unsqueeze(0).repeat(H, 1, 1),
            self.row_embed[:H].unsqueeze(1).repeat(1, W, 1),
        ], dim=-1).flatten(0, 1).unsqueeze(1)
        h = self.transformer(pos + h.flatten(2).permute(2, 0, 1),
                             self.query_pos.unsqueeze(1))
```

```
        return self.linear_class(h), self.linear_bbox(h).sigmoid()
```

待實現的功能

1. 實現 loss function：首先需要對輸出與 Ground Truth 建立一一對應，採用匈牙利演算法計算最小 loss，再用該 loss 值進行反向傳播，達到訓練的效果。

2. 使用 cocoapi 讀取資料集

3. 改變超參數

   (1) 改變 backbone 為 resnet50、resnet18...

   (2) 改變原始論文中 hidden_dim nheads num_encoder_layers num_decoder_layers 以及 embed 方式

   (3) 改變 loss function

4. 將結果可視化

   (1) 將 resnet 的提取出的局部特徵可視化

   (2) 將 transformer 自注意力關注的位置用熱點圖呈現

   (3) 將改變超參數後的結果用折線圖呈現

遇到的困難

這是我第一次建立模型，在 PyTorch 上有非常多常規的操作需要學習，例如 Tensor、optimizer、loss function 等等，另外 cocoapi 也是由於第一次使用而進度較緩慢。這些困難一一克服後，就只剩下如何使用 PIL 將結果呈現出來。

網路上只有使用 DETR 進行推論的參考範例，為了訓練我必須自己實作 loss function 與反向傳播的部分。

討論日期記錄

2/26、3/13、4/8、4/14

參考文獻

1. VASWANI, Ashish, et al. Attention is all you need. Advances in neural information processing systems, 2017, 30.

2. GIRSHICK, Ross. Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision. 2015. p. 1440-1448.

3. GIRSHICK, Ross, et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2014. p. 580-587.

4. LECUN, Yann, et al. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 1998, 86.11: 2278-2324.

5. COCO Dataset. [online]. 2017. COCO Consortium. Available from: http://cocodataset.org [Accessed 12 April 2024].

6. CARION, Nicolas, et al. End-to-end object detection with transformers. In: European conference on computer vision. Cham: Springer International Publishing, 2020. p. 213-229.