基于深度学习的物体检测 - 作业 8

peng00bo00

September 20, 2020

1. 本次作业是对 FaceBoxes 代码进行调试，现将代码核心内容总结如下：

   FaceBoxes 整体逻辑与 SSD 类似，核心组件包括 Backbone、PriorBox 和 MultiBoxLoss 三个部分，各组件的功能和训练流程如下：

   (a) Backbone 为 FaceBoxes 的主干，由快速消化网络模块 (RDCL) 和多尺度网络模块 (MSCL) 组成，用来产生人脸预测值：
      i. 输入图像经预处理后得到 1024×1024 尺寸的输入。
      ii. 输入图像送入 Backbone 经过快速消化网络模块 (RDCL) 和多尺度网络模块 (MSCL) 组成得到 3 个不同尺度的特征图，特征图尺寸分别为 32×32，16×16，8×8。
      iii. 在每张特征图上通过卷积运算得到锚框对应的类别 conf 和位置回归值 loc。

   (b) PriorBox 使用了锚框密集化的策略来产生不同尺度特征图上的锚框：
      i. 不同的特征图尺寸对应了不同的锚框大小：
         A. 32×32 尺寸特征图每个位置对应 32×32、64×64、128×128 三种不同大小的锚框；
         B. 16×16 尺寸特征图每个位置对应 256×256 大小的锚框；
         C. 8×8 尺寸特征图每个位置对应 512×512 大小的锚框；
      ii. 不同大小的锚框采用了不同的密集化策略：
         A. 32×32 大小的锚框每个位置对应 $4 \times 4 = 16$ 个锚框；
         B. 64×64 大小的锚框每个位置对应 $2 \times 2 = 4$ 个锚框；
         C. 其他大小的锚框每个位置对应 1 个锚框；
      iii. 因此，32×32 尺寸特征图每个位置对应 $16+4+1 = 21$ 个锚框，16×16 尺寸特征图每个位置对应 1 个锚框，8×8 尺寸特征图每个位置对应 1 个锚框。每张输入图像生成 $32 \times 32 \times 21 + 16 \times 16 \times 1 + 8 \times 8 \times 1 = 21824$ 个锚框。
      iv. 锚框的位置记录在 prior 中。

   (c) MultiBoxLoss 用来计算网络的分类和回归损失：
      i. 对每张图片将 GroundTruth 和 prior 根据 IoU 进行匹配，得到正样例锚框（人脸）和负样例锚框（背景）；
      ii. 使用 SmoothL1 损失函数计算全部正样例的位置损失 loss_l；
      iii. 对负样例锚框进行样本挖掘：
         A. 计算所有锚框的分类损失；
         B. 对于每张图片选择该图片对应的负样例锚框并按照锚框对应的分类损失大小进行排序；
         C. 按照正负样例比例 1:7 的比例，在每张图片上选择分类损失大的负样例和全部正样例锚框，将它们的分类损失求和作为最终的分类损失 loss_c；
      iv. 网络最终损失为 $L = 2 \times \text{loss\_l} + \text{loss\_c}$，得到总损失后即可利用优化器来更新网络参数。

   FaceBoxes 在不同数据集上的检测结果可参考 Fig.1-Fig.3。

Figure 1: AFW 检测结果

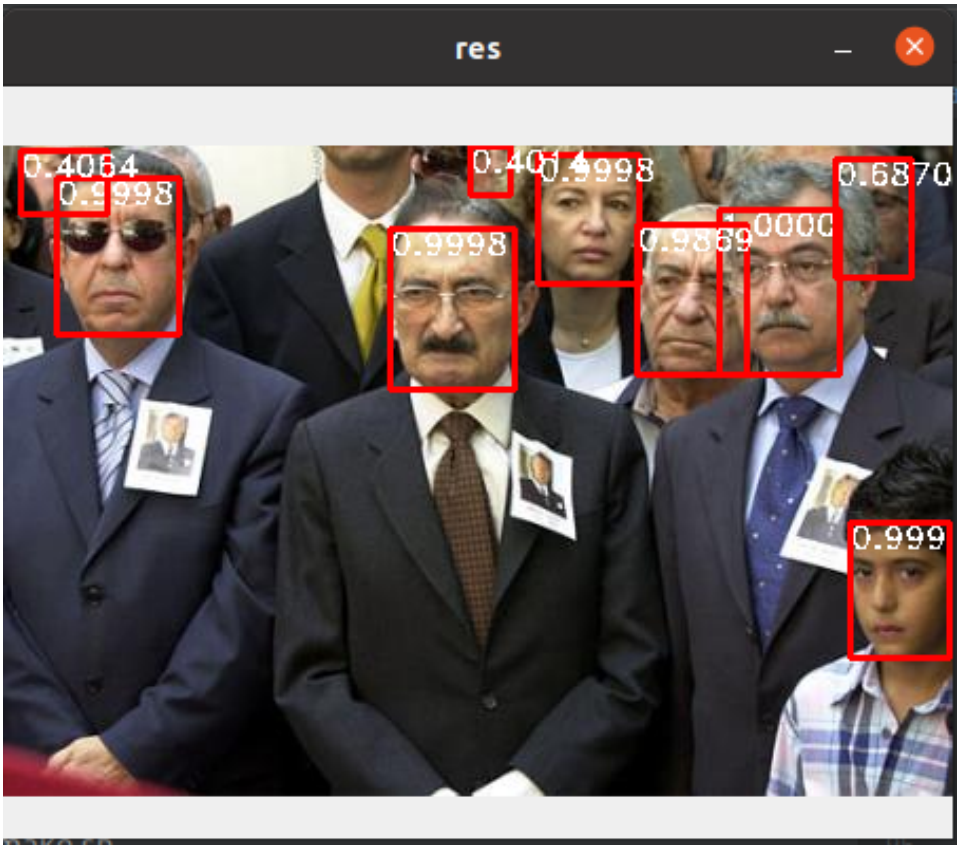

Figure 2: FDDB 检测结果

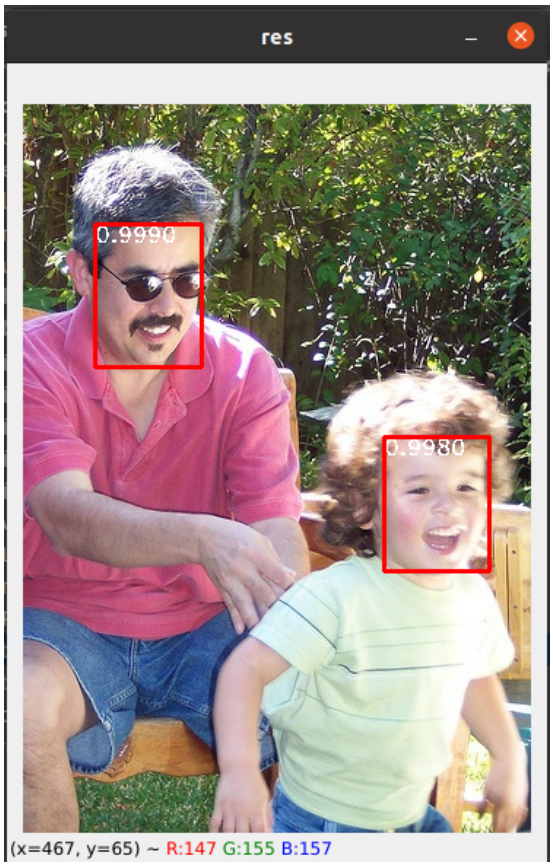

Figure 3: PASCAL 检测结果