Fast R-CNN

Summary

1. R-CNN is slow because each object proposal requires one forward pass and do not share computation.
2. Proposes sharing conv features b/t the obj proposal.

Architecture

1. An img is first processed by several conv layers and max pooling layers to produce a conv feature map.
2. Then, for each obj proposal, a region of interest (RoI) pooling layer extracts a fixed-length feature vector from the feature map.
3. The feature vector is passed through a sequence of fc layers which produce 2 output vectors per RoI:
   - softmax probabilities over K obj classes and a "background" class
   - per-class bb regression offsets.

RoI pooling layer

1. A RoI is a rectangular window into a conv feature map.
2. Each RoI is defined by a 4-tuple $(\underbrace{r, c}_{\text{top-left corner}}, \underbrace{h, w}_{\text{height \& width}})$.
3. The ROI pooling layer uses max pooling to convert the features inside the RoI into a small feature map with a fixed spatial dimension $H \times W$.
4. RoI max pooling works by:
   - dividing the $h \times w$ RoI window into an $H \times W$ grid of sub-window, each having size $\frac{h}{H} \times \frac{w}{W}$
   - max-pooling the values in each sub-window into the corresponding output grid cell.
   - pooling is applied independently to each feature map channel.

Loss Functions

1. For each RoI, the network produces two outputs:
   - $p = (p_0, \dots, p_K)$: softmax probabilities over $K+1$ classes.
   - $t^k = (t^k_x, t^k_y, t^k_w, t^k_h)$: bb regression offset for each of the K object classes.
     - specifies scale-invariant translation and log space $\frac{\text{height}}{\text{width}}$ shift relative to an obj proposal.

2. Each training RoI is labeled with a GT class u and GT bb regression target v.

3. The loss function is:

$$L(p, u, t^u, v) = L_{cls}(p, u) + \lambda [u \geq 1] L_{loc}(t^u, v)$$

   - $L_{cls}(p, u) = -\log p_u$
   - $[u \geq 1] = 1$ if $u \geq 1$ (the background class is labeled $u = 0$)
   - $L_{loc}(t^u, v) = \sum_{i \in \{x, y, w, h\}} \text{smooth}_{L_1}(t_i^u - v_i)$.
   - $\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$

Mini-batch Sampling

1. In each mini-batch, 25% of the RoI come from obj proposal with IoU overlap with GT bounding box.

2. The remaining 75% have maximum IoU in interval $[0.1, 0.5)$ (background class)

Scale Invariance

1. There are two methods they tested to achieve scale invariance:
    - brute force
    - image pyramids
2. The image pyramids method lead to slightly higher mAP.

Are more proposals always better ?

1. There are 2 methods to generate the set of obj. proposals:
    - sparse set (e.g. selective search).
    - dense set
2. They found that:
    - sparse set method leads to higher dense set method.
    - adding more proposals can lead to lower accuracy.
3. Average Recall (AR) is the SOTA for measuring obj proposal quality:
    - AR correlates well with mAP when using a fixed no. of obj proposal.
    - AR does not correlates well with mAP as the no. of proposals per image is varied.