Implementation of Relation Networks for Object Detection

Youcheng Li

[1]College of Artificial Intelligence, XJTU
West Xianning Road
Xi'an, Shaanxi, 710049,P.R. China
https://github.com/xjtulyc

Abstract

Although it is well believed for years that modeling relations between objects would help object recognition, there has not been evidence that the idea is working in the deep learning era. All state-of-the-art object detection systems still rely on recognizing object instances individually, without exploiting their relations during learning.

This work proposes an object relation module. It processes a set of objects simultaneously though interaction between their appearance feature and geometry, thus allowing modeling of their relations. It is lightweight and in-place. It does not require additional supervision and is easy to embed in existing networks. It is shown effective on improving object recognition and duplicate removal steps in the modern object detection pipeline. It verifies the efficacy of modeling object relations in CNN based detection. It gives rise to the first fully end-to-end object detector. Code is available at https://github.com/xjtulyc/Relation-Networks-for-Object-Detection

Review of Object Detection Pipeline

This work conforms to the region based object detection paradigm. The paradigm is established in the seminal work R-CNN and includes majority of modern object detectors. A four step pipeline is used in all previous works, as summarized here.

First step generates full image features. From the input image, a deep convolutional backbone network extracts full resolution convolutional features (usually 16× smaller than input image resolution). The backbone network is pretrained on ImageNet classification task and fine-tuned during detection training.

Second step generates regional features. From the convolutional features and a spare set of region proposals, a ROI pooling layer extracts fixed resolution regional features ($e.g.$, 7×7) for each proposal.

Third step performs instance recognition. From each proposal's regional features, a head network predicts the probabilities of the proposal belonging to certain object categories, and refine the proposal bounding box via regression. This network is usually shallow, randomly initialized, and jointly trained together with backbone network during detection training.

Last step performs duplicate removal. As each object should be detected only once, duplicated detections on the same object should be removed. This is usually implemented as a heuristic post-processing step called non-maximum suppression (NMS). Although NMS works well in practice, it is manually designed and sub-optimal. It prohibits the end-to-end learning for object detection.

In this work, the proposed object relation module is used in the last two steps. We show that it enhances the instance recognition and learn duplicate removal. Both steps can be easily trained, their independently or jointly. The joint training further boosts the accuracy and gives rise to the first end-to-end general object detection system.

Our implementation of different architectures to validate the effectiveness and generality of our approach, we experimented the different combination of state-of-the-art backbone networks (ResNet + FPN & MobileNet), and best-performing detection architecture (faster RCNN).

- Faster RCNN. It is directly built on backbone networks such as ResNet. RPN is applied on the conv4 feature maps. The instance recognition head network is applied on a new 256-d 1×1 convolution layer added after conv5, for dimension reduction. Note that the stride in conv5 is changed from 2 to 1, as common practice.

Object Relation Module

Given input set of N objects $\{(f_A^n, f_G^n)\}_{n=1}^N$, the relation feature $f_R(n)$ of the whole object set with respect to the n^{th} object, is compute as,

$$f_R(n) = \sum_m w^{mn} \cdot (W_V \cdot f_A^m) \tag{1}$$

The output is a weighted sum of appearance features from other objects, linearly transformed by W_V (corresponding to values V in Scaled Dot-Product Attention). The relation weight w^{mn} indicates the impact from other objects. It is computed as

$$w_{mn} = \frac{w_G^{mn} \cdot exp(w_A^{mn})}{\sum_k w_G^{kn} \cdot exp(w_A^{kn}} \tag{2}$$

Appearance weight w_A^{mn} is computed as dot product,

Copyright © 2022, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

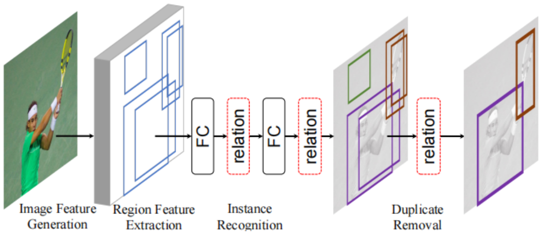


Figure 1: Current state-of-the-art object detectors are based on a four-step pipeline. Our object relation module (illustrated as red dashed boxes) can be conveniently adopted to improve both instance recognition and duplicate removal steps, resulting in an end-to-end object detector.

$$w_A^{mn} = \frac{dot(W_K f_A^m, W_Q f_A^n)}{\sqrt{d_k}} \tag{3}$$

Both W_K and W_Q and metrices and play a similar role as K and Q. They project the original features f_A^m and f_A^n into subspaces to measure how well they match. The feature dimension after project is d_k.

Geometry weight is computed as,

$$w_G^{mn} = max\{0, W_G \cdot \epsilon_G(f_G^m, f_G^n)\} \tag{4}$$

The geometry features of two objects are embedded to a high-dimensional representation, denoted as ϵ_G. The embedded feature is transformed by W_G into a scalar weight and trimmed at 0, acting as a ReLU non-linearity. The zero trimming operation restricts relations only between objects of certain geometric relationships.

An object relation module aggregates in total N_r relation features and augments the input object's appearance feature via addition,

$$f_A^n = f_A^n + Concat(f_R^1(n), ..., f_R^{N_r}(n)), foralln \tag{5}$$

Experiments

All experiments are performed on COCO detection datasets with 80 object categories. A union of 80k train images and a 35k subset of val images are used for training. Most ablation experiments report detection accuracies on a subset of 5k unused val images as common practice. For backbone network, we used ResNet-50 and MobileNet. The results are as follow Appendix.

In this section, we try to apply relation module in different stages of object detection and measure its performance using AP metrics.

Fig. 1 shows the object detection pipeline where relation modules are applied.

Comparing Different Backbones

In this section, we compare the performance between different backbone networks measured with AP. We test in Faster RCNN. The results are shown in Table 1. ResNet-50 with FPN is better than MobileNet. In the following comparing experiment, we choose ResNet-50 with FPN as the backbone network of the Faster RCNN.

Methods	mAP	mAP_{50}
ResNet-50	23.4	46.7
MobileNet	20.8	44.2

Table 1: Comparing Different Backbones

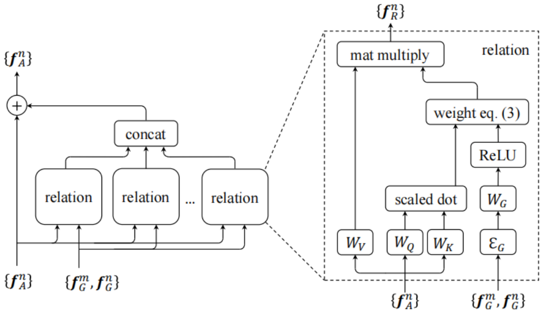


Figure 2: Left: object relation module; Right: relation feature.

Relation for Instance Recognition

The architecture of instance detection step is shown in fig. 2. The results shown in Table 2 shows that the Relation Module can improve the performance.

Relation for Duplicate Removal

The architecture of instance detection step is shown in fig. 3. The results shown in Table 3 shows that the Relation Module can improve the performance.

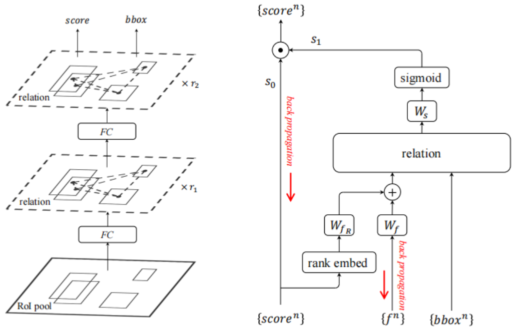
Relation for Object Detection

Finally, we add Relation Module in the third step and the forth step of object detection pipeline like fig. 1. The results shown in Table 4 shows that the Relation Module can improve the performance.

Thank you for reading this report carefully.

Methods	mAP	mAP_{50}
Relation	26.2	48.1
no Relation	23.4	46.7

Table 2: Relation for Instance Recognition



(a) enhanced *2fc* head (b) duplicate removal network

Figure 3: Illustration of enhanced 2fc head (a) and duplicate classification network (b) by object relation modules.

Methods	mAP	mAP_{50}
Relation	25.9	47.2
no Relation	23.4	46.7

Table 3: Relation for Instance Recognition

Methods	mAP	mAP_{50}
Relation	28.1	48.2
no Relation	23.4	46.7

Table 4: Relation for Object Detection