Learning Globally Optimized Object Detector via Policy Gradient

Yongming Rao[1,2,3], Dahua Lin[4], Jiwen Lu[1,2,3], Jie Zhou[1,2,3]
[1]Department of Automation, Tsinghua University
[2]State Key Lab of Intelligent Technologies and Systems, Tsinghua University
[3]Beijing National Research Center for Information Science and Technology
[4]Department of Information Engineering, The Chinese University of Hong Kong
raoyongming95@gmail.com; dhlin@ie.cuhk.edu.hk; {lujiwen, jzhou}@tsinghua.edu.cn

Abstract

In this paper, we propose a simple yet effective method to learn globally optimized detector for object detection, which is a simple modification to the standard cross-entropy gradient inspired by the REINFORCE algorithm. In our approach, the cross-entropy gradient is adaptively adjusted according to overall mean Average Precision (mAP) of the current state for each detection candidate, which leads to more effective gradient and global optimization of detection results, and brings no computational overhead. Benefiting from more precise gradients produced by the global optimization method, our framework significantly improves state-of-the-art object detectors. Furthermore, since our method is based on scores and bounding boxes without modification on the architecture of object detector, it can be easily applied to off-the-shelf modern object detection frameworks.

1. Introduction

Object detection is one of the oldest and most fundamental tasks in computer vision. Compared to the object recognition task, object detection is more challenging because it requires accurate localization and classification of multiple objects at the same time. To tackle this problem, most modern object detectors [5, 6, 16, 18, 20, 22] are trained through a reduction that converts object detection into a multi-task learning of object classification and localization for each object *independently*. This reduction introduces a gap between learning and inference, where the process of learning only needs to evaluate each possible object proposal (RoIs in [5], anchors in [16, 18, 21, 22]) and feed back to the detector, but the process of inference is supposed to perform an optimal selection of redundant candidates to obtain accurate detection results, which is a global optimization problem with a constraint on the amount of detections (for instance, 100 for COCO evaluation [17]).

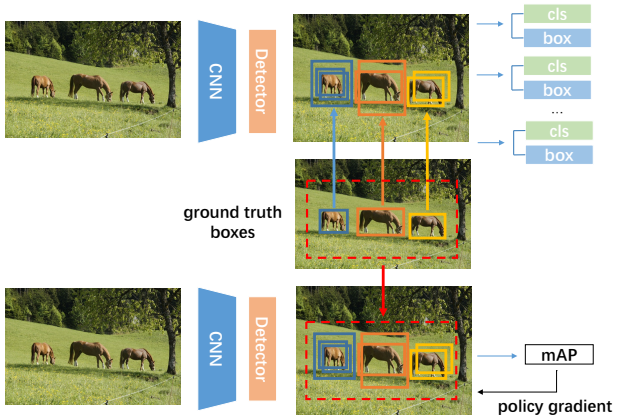

Figure 1. The key idea of our approach. We propose an end-to-end framework for learning a globally optimized object detector by using the policy gradient method. The figure presents the differences between the existing object detection framework (on the top) and our proposed framework (on the bottom). Existing object detection framework assigns each ground truth box to proposals that has the high IoU value and trains each detection candidate *independently*. By employing the policy gradient method, our framework supervises boxes by using the *global* information and directly improves mAP. Red box represents the set of bounding boxes.

In [5, 6, 16, 18, 20, 22], a method called non-maximum suppression (NMS) has been used to bridge the gap. NMS performs a test-time post-processing to merge bounding boxes that might belong to the same object. This method greedily selects high scoring bounding boxes and eliminates near-by bounding boxes that have lower scores since they are likely to cover the same object. NMS is built on the assumption that more accurate detection candidates have higher confident scores, which is not explicitly designed in the objective of detector training in previous methods. Unsurprisingly, both our survey and experiments (see Sec-

tions 2 and 4.1) show that the assumption does not hold in most cases, especially in some challenging datasets such as COCO. The inconsistency between learning and inference will harm the detection performance. Therefore, it is important to design a framework that closes the gap in the training phase.

Aside from developing richer and hierarchical features to get better performance [12, 15, 18], better objective functions have also been proven to be effective on improving object detection results in most recent works [16, 25]. Both Focal Loss [16] and Online Hard Example Mining (OHEM) [25] focus on class imbalance, which may cause inefficient training because the standard cross-entropy loss cannot provide useful supervision signals. Both of these two methods adopt a re-weighting scheme to address this problem. However, improperly assigned labels can lead to misleading gradient, which may be amplified by gradient re-weighting schemes and cause collapse during training. By utilizing global information of detection candidates, a generic solution to both the class imbalance and improper label problem can be proposed, which can further improve the detection performance.

To address the above-mentioned challenges, we propose an end-to-end training framework for object detection, which attempts to learn globally optimized object detectors by employing a policy gradient method. Our method is effective yet simple to implement, which is a simple modification to the standard cross-entropy gradient inspired by the REINFORCE algorithm [31] in reinforcement learning, as shown in Figure 1. In our approach, the cross-entropy gradient is adaptively adjusted according to overall mean Average Precision (mAP) of the current state for each object proposal, which leads to more effective gradients and global optimization of detection results. Moreover, no computational overhead is introduced. Benefiting from more precise gradient produced by the global optimization in our method, our framework significantly improves state-of-the-art object detectors, such as Faster R-CNN [22] and Faster R-CNN with Feature Pyramid Networks [15] (FPN). Without tricks, our method improves the COCO-style mAP by 2.0% over a strong baseline of Faster R-CNN and 1.8% over Faster R-CNN with FPN on the COCO object detection dataset. Furthermore, since our method is based on the confident score and bounding box regression without modification on the architecture of object detector, it can be easily applied to off-the-shelf modern object detection frameworks.

2. Related work

Benefiting from better features, the past few years have witnessed a great development in object detection [6, 8, 15, 22]. Compared to other high-level image understanding tasks in computer vision, such as semantic segmentation [19], it is more difficult to design an end-to-end solution for object detection. Consequently, region-based methods have been widely used for accurate object detection, which reduces the problem of object detection to many *independent* subproblems of evaluating dense or sparse regions. However, utilizing global information in object detection has rarely been explored in recent works.

CNN-based Object Detectors: Convolutional neural networks (CNN) was first introduced to the field of object detection by Girshick *et al.* [6], which almost doubled the detection performance on PASCAL VOC datasets. The method, called R-CNN, employs CNN for localization by performing a region-based recognition operation, which decomposes images with multiple objects into several pre-extracted regions of interest (RoIs). Fast R-CNN [5] and Faster R-CNN [22] further develop and accelerate the method by sharing CNN feature and combining CNN-based region proposal networks, respectively. Different from R-CNN-style detectors which use a propose-refine pipeline to achieve highly accurate detection results, one-stage object detectors [16, 18, 20] aim to achieve real-time detection applications. Most one-stage detectors use an anchor-refine pipeline, where anchors are fixed on each pixel of feature maps. In all of the above-mentioned object detectors, proposals are treated independently in the training phase.

Objective Function of Object Detection: In Fast R-CNN and its descendants, a multi-task loss of classification and regression is used for region-based recognition. Labels of region proposals are assigned according to a hard threshold criterion. Since the number of foreground RoIs and background RoIs are usually imbalanced and varied during training, vanilla objective function may produce useless or misleading learning signal, thus training at most positions is inefficient. To address the problem of class imbalance, Online Hard Example Mining [25] and Focal Loss [16] are proposed to improve the training of sparse and dense object detection respectively. Previous works show that better objective functions will significantly improve detection performance.

Modifications on Non-maximum Suppression: There have been some efforts to modify NMS for better detection results. For example, Hosang *et al.* proposed a parametric auxiliary model Gnet [10] to integrate information among neighboring bounding boxes to re-score bounding boxes after the standard pipeline of object detectors. Gnet focuses on penalizing double detections, which reduces redundant boxes. On the contrary, another modification on NMS, soft-NMS [1], tries to keep boxes that are eliminated during performing NMS and rescore these boxes. Both Gnet and soft-NMS improve the final detection results, which indicates that detections are both redundant and inaccurate and the assumption of NMS does not hold in most cases. Different from these works, our method does not change the NMS method but employs a smooth and more precise objective

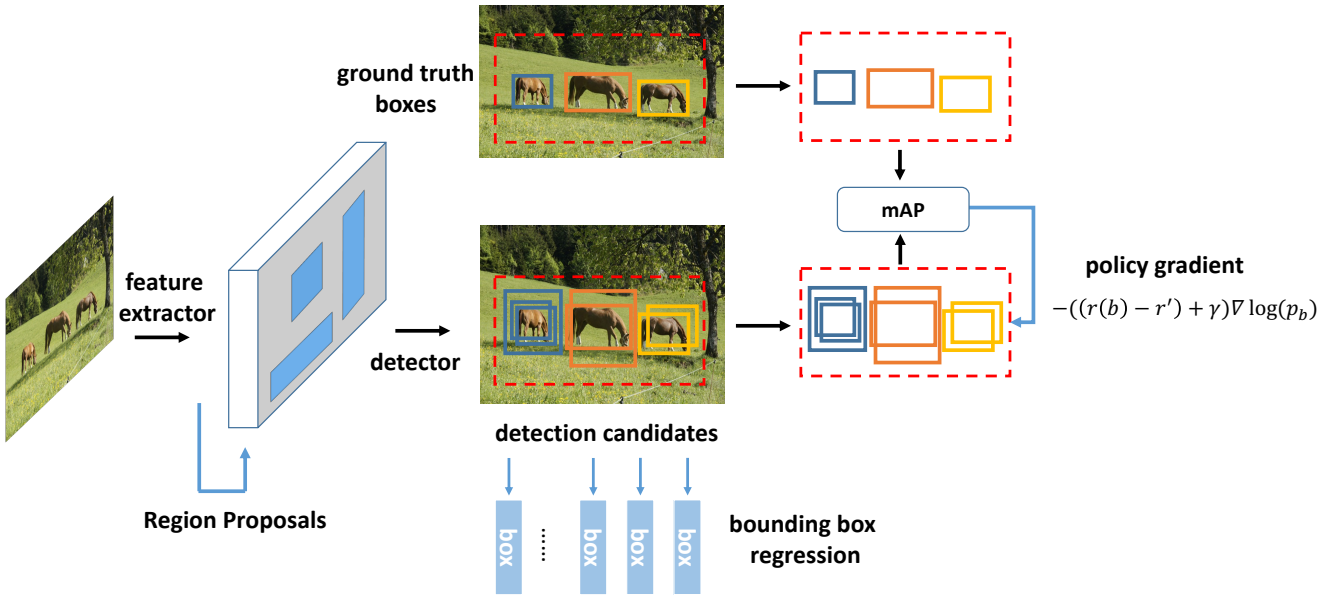

Figure 2. The overall framework of proposed approach. The goal of learning global optimized object detector is to maximize the mAP between detection results and ground truth boxes. Object detector are trained by two supervision signals: a smooth L1 loss function for bounding box regression and a policy gradient for the global optimization.

function to train object detectors, which treats NMS as a part of the detector.

End-to-end Training for Object Detector: Since CNN was introduced to the field of object detection, there has been much interest in designing an end-to-end learning framework for object detection. Faster R-CNN [22] combines the region proposal module to the pipeline of object detection, where a trainable region proposal networks (RPN) was proposed and significantly improves the quality of region proposals. [10] and [9] merge NMS into the end-to-end learning framework for modern object detectors. However, all of these efforts focus on utilizing local information of proposal boxes, which attempts to address the problem of double detections and occlusion. Although some efforts have been made on post-processing detection results such as [2, 24], global planning of detection results for modern object detector has not been visited yet.

3. Approach

The key idea of global optimization for object detection is presented in Figure 1. Existing learning frameworks train each detection candidate independently with a multi-task loss of classification and localization. Global optimization method supervises detector using the policy gradients from the differences between detection results set and ground truth bounding boxes set, where the global information between objects can be utilized.

3.1. Overview of Faster R-CNN

We start by reviewing the Faster R-CNN [22] detector breifly, on which our framework is built.

There are two stages in Faster R-CNN, which are the Region Proposal Network (RPN) and Fast R-CNN head. RPN consists of several convolutional layers and produces confident scores and bounding boxes for each reference box, called anchor. Anchors that have different scales and aspect ratio are fixed at each position of feature maps, where a dense and class-agnostic prediction is performed to find object proposals. Fast R-CNN head extracts feature using RoIPool operation for each RoI, and performs classification and bounding box regression. These two stages share the same features for fast inference.

During training, a multi-task learning scheme with a cross-entropy loss for classification and a smooth L1 loss for localization is used, which can be formulated as:

$$L = L_{cls} + \alpha L_{loc}. \tag{1}$$

where α is a balancing weight between classification and localization. After labels are assigned to each RoI, the multi-task learning procedure is performed for each RoI *in parallel*. We argue that the relation information between RoIs that is ignored in Faster R-CNN can be further utilized to improve object detectors.

3.2. Problem Formulation

Given an image I, object detection is a task that maps the image to a set of bounding boxes with class information $B = \{(b_i, y_i)\}$, where the number of bounding boxes equals to the number of objects of interest and bounding boxes are the minimum enclosing rectangles of corresponding objects. In most modern object detectors, a redundant set of detection candidates B' are provided, which means

$$|B'| \geq |B|. \tag{2}$$

Evaluation of bounding box detection task does not require object detector to provide a set of bounding boxes that contains exact $|B|$ detections, on the contrary, the number of objects in the image is agnostic for object detector. Therefore, a constraint on the amount of detections is used in both COCO benchmark [17] and PASCAL VOC [3].

The objective of object detection can be formulated as

$$\begin{aligned} &\mathcal{H} = \max_\theta \text{mAP}(F_\theta(I), B), \\ \text{subject to} \quad &|F_\theta(I)| \leq N_{bb} \end{aligned} \tag{3}$$

where F is the object detector that takes an image as input and produces a set of detection candidates, θ is the set of parameters in F and N_{bb} is the threshold of maximum detection candidates. mAP the evaluation function used in both [17] and [3], which computes mean Average Precision between detection candidates and ground truth boxes with a fixed threshold (0.5 for PASCAL VOC) or a range of threshold ([0.5:0.95] in 0.05 increments for COCO benchmark).

As described in the previous section, detection systems are traditionally trained using the multi-task loss to optimize each detection candidate independently. In order to utilize the global information in object detection, the set of detection candidates are jointly processed in our proposed method, which aims to maximum mAP between detection candidates and ground truth boxes. Therefore, the objective of learning globally optimized object detector can be written as follows:

$$L(\theta) = -\mathbb{E}_I(\text{mAP}(F_\theta(I), B_I)) + \lambda \parallel \theta \parallel^2 \tag{4}$$

where $\mathbb{E}_I$ is the expectation over all training images and the second term is the regularization term with a positive regularization factor λ.

3.3. Global Optimization for Object Detector

Since non-maximum suppression is a hard selection operation on detection candidates, the optimization problem described in equation 4 cannot be solved by standard gradient decent algorithm. To tackle this problem, we propose a policy gradient based method inspired by REINFORCE algorithm [31] to address the global optimization problem for object detection. Detectors such as Faster R-CNN can be viewed as an *agent* that interacts with an external *environment* , in our case, images. The aim of the agent is to get maximum possible mAP between selected detection candidates and ground truth boxes, which can be regarded as the *reward* to the agent. The parameters of the object detectors, θ, define a policy p_θ, which results in an *action* that is used to select detections. After each action, the agent presents a set of detection candidates and observes a reward from evaluation system. Unlike reinforcement learning problems that are based on sequences of decisions such as robot control and video games, *states*, in our case, features of images, consist of descriptions of the environment, which are not determined by the previous states or actions. Strictly speaking, the above formulation is not a *full* reinforcement learning framework, which is beyond our scope in this paper. Here we concentrate in employing policy gradient method as a gradient approximation tool to utilize global information and improve the procedure of training object detector.

In summary, the goal of end-to-end training is to minimize the negative expected evaluation metric, i.e., mean average precision (mAP):

$$L(\theta) = -\mathbb{E}_I(r_I) = -\mathbb{E}_I(\text{mAP}_I) \tag{5}$$

Policy Gradient for Global Optimization: For object detection task, expected reward r is non-differentiable because of non-maximum suppression. In order to compute $\nabla L(\theta)$ directly, we can use a policy gradient method, which computes the expected gradient of the non-differentiable reward function as follows:

$$\nabla L_I(\theta) = -\mathbb{E}_a[r(a)\nabla_\theta \log(p_a)] \tag{6}$$

where **action** a can be defined as "selecting a set of bounding boxes from all candidates". If there are m candidates c categories and n boxes selected at each time, the number of all possible actions will reach $\#a = \binom{m \times c}{n}$, which can be very large (for instance, if $m = 1000$, $n = 10$ and $c = 80$, $\#a = 3 \times 10^{42}$). Exploring such a large action space is hard and inefficient.

To apply objective in Equation 6 to the training procedure of object detector, we use some approximation tricks to reduce the size of action space and obtain more efficient gradient. Since our proposed method is applied on the pre-trained object detectors, we can assume that the pre-trained detector can correctly produce categories of input regions:

$$p_b = P(b, l) \gg P(b, l'), \forall l' \neq l, \tag{7}$$

where $P(b, l)$ is the softmax output of bounding boxes b at category l. Therefore, each bounding box can be assigned to a ground truth category and will not be sampled as other categories during the policy gradient training. p_b is the probability that detector recognizes b as the ground truth class, i.e, score of b.

p_a is the probability of action a, therefore, $p_a = \prod_{b \in a} p_b$. We can further simplify Equation 6 as:

$$\begin{aligned} & \mathbb{E}_a[r(a)\nabla_\theta \log(p_a)] \\ = & \sum_a p(a) r(a) \nabla_\theta \log(\prod_{b \in a} p_b) \\ = & \sum_a [p(a) r(a) \sum_{b \in B'} [\delta(a,b) \nabla_\theta \log(p_b)]] \\ = & \sum_{b \in B'} [\nabla_\theta \log(p_b) \sum_a [p(a) r(a) \delta(a,b)]] \end{aligned}$$

we can define $r(b) = \sum_a [p(a) r(a) \delta(a,b)]$. Note that summing up all possible actions is impractical, so we sampled $k = 10$ actions according to p_a stochastically to compute $r(b)$ in our implementation. We find that sampling several actions during a single gradient calculation, instead of sample *one* action per iteration as typical implementations in previous policy gradient methods, is more efficient and will significant stabilize the training procedure. Therefore, expected policy gradient can be written as:

$$\nabla L_I(\theta, b) \approx -r(b) \nabla_\theta \log(p_b). \tag{8}$$

Policy Gradient with a Baseline: Baseline is widely used in policy gradient based algorithm to reduce the variance of the gradient estimation [23, 27]:

$$\nabla L_I(\theta, b) \approx -(r(b) - r') \nabla_\theta \log(p_b), \tag{9}$$

where the baseline r' can be chosen arbitrarily [31] as long as it does not depend on the action a. Adding a baseline term will not change the expectation of the gradient, since baseline is irrelevant with confident score p_b. Here we choose the mAP of detection results obtained by performing NMS as the baseline. Intuitively, we can see that bounding boxes that have higher IoU with ground truth boxes than selected ones will get positive rewards, while actions of selecting low IoU boxes will be penalized. Besides, we set the rewards of boxes that have already been selected to zero.

Joint Training with Cross-entropy Loss: To make the assumption in Equation 7 hold during training, an auxiliary cross-entropy loss can be used to further utilize the annotation information and stabilize training. We can summarize the formulation of final gradient as:

$$\nabla L_I(\theta, b) \approx -((r(b) - r') + \gamma) \nabla_\theta \log(p_b), \tag{10}$$

where γ is the weight of cross-entropy loss. We can compute derivatives as follows:

$$\frac{\partial L_I(\theta, b)}{\partial x} \approx ((r(b) - r') + \gamma)(p_b - 1) \tag{11}$$

where x is the input to the softmax function. Policy gradient can be regarded as an adjustment of cross-entropy loss, which dynamically adjusts the final gradient according to the current detection results for each detection candidate. It can be observed that the key elements of our method include: (1) ensuring that better detection candidates have higher confident scores, (2) making detector aware of the NMS during training phase and (3) amplifying the gradient of hard examples, which are the missing pieces in traditional detector learning.

Algorithm 1 Learning Global Optimized Object Detector

Input: A Set of image $\{I\}$, corresponding ground truth bounding boxes $\{B\}$
Output: object detector F_θ

initialize F_θ with ImageNet pre-trained model or train F_θ in normal way
for $i \leftarrow 1, 2, ..., M$ **do**
  Sample random minibatch from $\{I\}$
  Compute RoIs or anchors for image I
  Assign ground truth boxes to RoIs or anchors
  Compute detections candidates and results after NMS
  Compute smooth L1 loss L_{bb} for bounding box regreesion
  Compute policy gradient g_{cls} according to equation 11
  Update θ with gradient $\nabla_\theta L_{bb} + g_{cls}$
end for
return object detector F_θ

The overall framework of our proposed approach is summarized in Figure 2. Algorithm 1 details the training procedure of the proposed method.

4. Experiments

4.1. Implementations

We evaluate our proposed method on the challenging 80 category COCO detection dataset [17]. All of our models are trained using the union of 80k train images and a 35k subset of val images (`trainval35k`). We report the standard COCO metrics including COCO-style mAP, mAP_{50}, mAP_{75}, mAP_S, mAP_M and mAP_L on a 5k subset of val images (`minival`) [1].

Following [7, 15, 16], we use the networks that are pre-trained on the ImageNet1k classification dataset [13] as backbone networks and fine-tune our models on the object detection dataset. All of our models are trained based on ResNet-101 model that is provided by authors of [8].

Our proposed method is a generic solution for training object detector. In this section, we adopt our method in

[1] We use the same split as used in https://github.com/rbgirshick/py-faster-rcnn

Detection model	training method	greedy NMS	soft NMS	mAP	mAP_{50}	mAP_{75}	mAP_S	mAP_M	mAP_L
Faster R-CNN	standard	✓		36.3	57.3	38.8	17.7	42.4	51.4
Faster R-CNN	standard		✓	36.9	57.2	40.1	18.0	42.7	52.1
Faster R-CNN	OHEM	✓		36.9	57.3	40.2	17.7	42.7	52.4
Faster R-CNN	ours ($\gamma = 0$)	✓		37.6	60.0	40.2	19.6	42.6	52.0
Faster R-CNN	ours ($\gamma = 1$)	✓		38.3	60.6	40.9	20.7	43.2	52.6
Faster R-CNN	ours ($\gamma = 1$)		✓	**38.5**	**60.8**	**41.3**	**20.9**	**43.4**	**52.7**
Faster R-CNN with FPN	standard	✓		37.7	58.5	40.8	19.3	41.7	**52.3**
Faster R-CNN with FPN	ours ($\gamma = 1$)	✓		**39.5**	**60.2**	**43.3**	**22.7**	**44.1**	51.9

Table 1. Ablation experiments for our proposed method, evaluated on COCO `minival` set. All models are trained on `trainval35k` set. Standard denotes that training object detector use the combination of cross-entropy loss and smooth L1 loss. All results are based on ResNet-101 network and share the same hyper-parameters. We can see that our proposed method improves COCO-style mAP by 2.0 points over a strong baseline of Faster R-CNN and 1.8 points over Faster R-CNN with FPN.

state-of-the-art object detectors, such as Faster R-CNN [22] and Faster R-CNN with Feature Pyramid Networks [15] (FPN) to evaluate our method. To demonstrate the effectiveness of our method, we make some modifications to the original implementation to build stronger baselines for object detection. Although our baseline models significantly outperform original models reported in pervious papers [15, 22], our method can further improve the detection performance greatly.

Object Detection with Faster R-CNN: Faster R-CNN is a combination of region-based object detector (Fast R-CNN) and Region Proposal Networks (RPN), which can be end-to-end trained from image input to detection candidates. Convolutional neural networks are used to produce image features, which are shared by both object detector and region proposal network for faster inference.

The original Faster R-CNN models are built on the VGG-style networks [28], which are relatively shallow and inefficient compared to prevalent deeper architectures such as ResNet [8] and its descendants [30, 32]. Following [8, 15], we use the ResNet network of depth 101 layers (ResNet-101) as the backbone architecture for feature extraction in our experiments. The base ResNet-101 model is pre-trained on ImageNet1k dataset as initialization for object detection task. During training phase, layers in first two stages of ResNet-101 and all batch normalization layers are freezed for more stable training. Different from the original implementation of Faster R-CNN with ResNets [8], which extracts features from the final convolutional layer of the 4-th stage, the whole model with dilated convolution layers is used for deeper and larger features, as [14]. In our implementation, features are extracted from the last convolutional layer of ResNet, i.e., the final convolutional layer of the 5-th stage, and convolutional layers in the 5-th stage are replaced by dilated convolutional layers to reduce the feature stride and maintain the field of view simultaneously [19]. To make the best use of ImageNet pre-trained model, the weights of dilated layers are initialized by corresponding convolutional layers.

Region Proposal Network takes image features as input and outputs a set of object proposals with confident scores. We use a convolutional layer with kernel size 3×3 to map image features from ResNet to a 512-d feature map. Two branches of classification and localization are used to produce $2A$ scores and $4A$ coordinates respectively for each point at feature map with two 1×1 convolutional layers as described in [22], where A is the number of anchors. Following [15], anchors have areas of $\{32^2, 64^2, 128^2.256^2, 512^2\}$ pixels with multiple aspect ratios of $\{1:2, 1:1, 2:1\}$ for each scale are used in RPN. Thus we set A to 15 in our implementation. During training, labels of anchors are assigned according to their IoU with ground truth boxes. An anchor is assigned a positive label if it has an IoU with ground truth boxes higher than 0.7, and assigned a negative label if it has an IoU with ground truth boxes lower than 0.3. Note that anchors that have the highest IoU with ground truth boxes are also assigned positive labels and the rest of anchors are ignored during training. During training and inference, RPN produces region proposals after performing NMS with threshold 0.7 on all anchors. Proposals are eliminated if their shorter sizes are less than 2 pixels, different from 16 pixels in previous works. Because we find it is important to keep small proposals, which can greatly improve the detection performance on small objects (mAP_S).

For each region proposal, a RoIAlign operation introduced in [7] is performed to extract a position sensitive and fixed size feature map from image feature. Unlike [7] using a 7×7 RoIAlign operator, we find denser sampling (*e.g.*, 14×14, as [11]) will extract more information for large objects and feed back more information to backbone CNN, which is helpful to get more precise predictions and fine-tune backbone network more sufficiently. Since the 5-th stage of ResNet has been used for feature extraction, we apply an extra 3×3 convolutional layer with stride 2 to produce a size of $(7, 7, 256)$ feature for each RoI, followed by

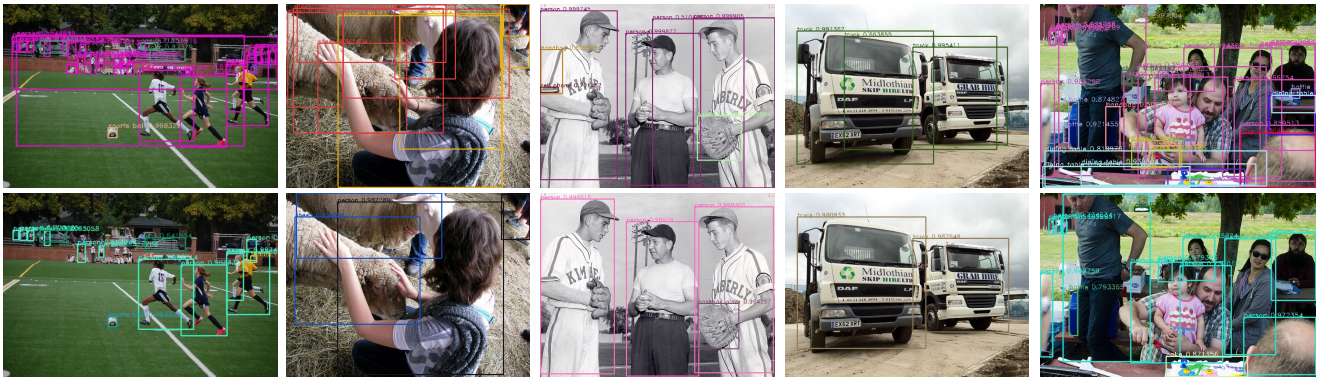

Figure 3. Baseline model (top) vs. globally optimized object detector (bottom, Faster R-CNN model). We only keep the boxes with the confident scores higher than 0.5. We can see that wrong detections with high confident scores can be barely found in results of our model.

two 1024-d fully connected layers before the final classification and bounding box regression layer that is used in [15] to form a head architecture of detector. Each of the above layer is followed by a ReLU activation, and dropout [29] is not used. Note that compared to conv5 head [8] and 2-fc MLP head [15], our head architecture has less parameters ($\sim$ 33% compared to 2-fc MLP for faster R-CNN) and GPU memory usage.

Object Detection with Feature Pyramid Networks: FPN is a top-down architecture with lateral connections to build high-level semantic feature maps for several scales, which can significantly improve the detection results for objects at vastly different scales. We adopt this design to build a stronger baseline than Faster R-CNN and evaluate the generality of our proposed method.

Following [15], we build a feature pyramid upon the backbone ResNet model by using features from final convolutional layer of each stage. Since dilated layers are used in the final stage of ResNet, only features from the 2-nd, 3-rd and 5-th stage are utilized, which are denoted by $\{C_2, C_3, C_5\}$. We apply a 1×1 convolutional layers on each of these feature maps, followed by lateral connections to produce 256-channel feature maps at different scales $\{P_2, P_3, P_5\}$ and apply a 3×3 convolutional layers with stride 2 on C_5 to produce the coarsest resolution map P_6. Anchors that have areas of $\{32^2, 64^2, 128^2\}$ are assigned to $\{P_2, P_3, P_5\}$ respectively, and anchors that have areas of $\{256^2, 512^2\}$ are assigned to P_6. We find that mini-batch is mainly composed of small RoIs during training, thus samples that are assigned to coarse feature maps are usually sufficient. Merging large anchors to P_6 can improve the detection results on large objects (mAP_L).

Optimization: We follow the *image-centric* sampling strategy [5, 22] to train our models. All input images are resized such that their shorter side is 800 pixels, following [7, 15]. We adopt synchronized SGD training on 8 GPUs. Each mini-batch has 2 images per GPU. For each image, 256 anchors are sampled for training RPN and 512 RoIs [7] are sampled with a ratio of 1:3 of positive and negatives [5, 7] for training detector. We use a weight decay of 0.0001 and a momentum of 0.9. For Faster R-CNN, we train our models for 80k iteration, with a learning rate of 0.01 which is decreased by 10 at the 50k iteration and 65k iteration. For Faster R-CNN with FPN, we train our models for 160k iteration, with a learning rate of 0.01 which is decreased by 10 at the 100k iteration and 130k iteration. For global optimization learning, additional 22.5k iteration training using our proposed method is performed on baseline object detection model with a learning of 0.001 which is decreased by 10 at the 15k iteration. During training, we compute mAP over each image to produce the policy gradient described in Section 3.3. In our implementations, mAP is calculated at multiple (0.5:0.95) IoU threshold following COCO evaluation, where we computes per-instance precision at multiple IoU thresholds and then average them to obtain AP of the input image. Other training details are as in [22].

Inference: We evaluate all our models using single-view testing. At test time, we set the number of region proposal before and after performing NMS to 8000 and 1000 respectively. The threshold of final NMS is set to 0.5, which is consistent with training.

4.2. Ablation Experiments

To investigate the performance of our proposed method, we conducted several ablation experiments. Experiment results are presented in Table 1.

Comparisons with baselines: For fair comparisons with baseline models without global optimization training, global optimization learning is performed on the baseline object detection models, where their performance cannot be further improved by continuing training. Both baseline

Model	backbone	test set	mAP	mAP_{50}	mAP_{75}	mAP_S	mAP_M	mAP_L
SSD [18]	ResNet-101	`test-dev2015`	26.8	46.5	27.8	9.0	28.9	41.9
DSSD [4]	ResNet-101	`test-dev2015`	33.2	53.3	35.2	13.0	35.4	51.1
Faster R-CNN with FPN [7]	ResNet-101	`test-dev2015`	37.3	59.6	40.3	19.8	40.2	48.8
RetinaNet [16]	ResNet-101	`test-dev2015`	39.1	59.1	42.3	21.8	42.7	50.2
Faster R-CNN with FPN [15]	ResNet-101	`minival`	35.2	58.2				
Faster R-CNN with TDM [26]	Inception-ResNetv2	`minival`	38.1	58.6	40.7	17.4	41.1	54.7
Faster R-CNN (ours)	ResNet-101	`minival`	38.3	60.6	40.9	20.7	43.2	52.6
Faster R-CNN with FPN (ours)	ResNet-101	`minival`	39.5	60.2	43.3	22.7	44.1	51.9

Table 2. Object detection single model results. We compare results of our proposed method with state-of-the-art models, evaluated on COCO `minival` set.

models and our proposed models are trained using the same hyper-parameters with the proposed modifications, following above-mentioned details.

We can see that our proposed method improves mAP of Faster R-CNN detector by 2.0 points over a strong baseline of 36.3 and improves Faster R-CNN with FPN model by 1.8 points over the baseline of 37.7 under the greedy NMS setting. In addition, the performance on small objects (mAP_S) and accurate detection (mAP_{75}) is boosted by large margins of 3.0 and 2.1 over Faster R-CNN, 3.4 and 2.5 over Faster R-CNN with FPN, respectively, which is the core problem that we want to tackle using the idea of global optimization. In summary, our proposed method greatly improves the baseline models. Experiment results demonstrate the effectiveness of our method. Examples of detection results are shown in Figure 3.

Comparisons with other methods: We compared our proposed method with other modifications on objective function and NMS using the Faster R-CNN model.

We first compared our method with OHEM [25], which achieved 36.9 mAP and improved the baseline model by 0.6 points. We can see that our model outperforms OHEM model by 1.4 points, showing that our method is more effective than OHEM for training sparse detectors.

Furthmore, we tested our models by performing soft NMS [1] during inference. Soft NMS is another attempt that aims to close the gap between learning and inference. It can be observed that soft NMS achieves similar results with OHEM, which improves the baseline model by 0.6 mAP. We note that soft NMS improves our proposed model less than the baseline model by 0.4 points ($\sim 66\%$ of improvement), which indicates that our method can further close the gap between the phase of training and inference.

How important is joint learning? To understand the importance of joint learning of policy gradient and cross-entropy loss, we trained a model using policy gradient without cross-entropy term ($\gamma = 0$). We argue that adding the term of cross-entropy can further utilize the annotation information and stabilize training process. Results show that this modification can improve the final results by 0.6 mAP. Note that even without cross-entropy loss, policy gradient can keep the learning process stable and improve the baseline model by 1.3 points. This result suggests that policy gradient plays a more important role in our method.

4.3. Comparisons with State-of-the-art

We evaluated our proposed method on the bounding box detection task of the challenge COCO dataset and compare our results to recent state-of-the-art methods Results are presented in Table 2 for our models that are based on Faster R-CNN and Faster R-CNN with FPN baselines. Compared to existing methods, our approach achieves a very competitive results in COCO dataset. Our Faster R-CNN with FPN model outperforms other single model results on `minival` set.

5. Conclusion

In this paper, we have presented a new method for learning globally optimized object detector via policy gradient, which is a simple yet effective modification to standard cross-entropy gradient.

While our method is designed for Faster R-CNN like object detectors, it can also be easily applied to off-the-shelf modern object detection frameworks, which is an interesting future work.

Acknowledgements

This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB1001001, by the National Natural Science Foundation of China under Grant 61672306, Grant U1713214, Grant 61572271, and Grant 61527808, in part by the National 1000 Young Talents Plan Program, in part by the National Basic Research Program of China under Grant 2014CB349304, in part by the Shenzhen Fundamental Research Fund (Subject Arrangement) under Grant JCYJ20170412170602564.

References

[1] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis. Improving object detection with one line of code. *arXiv preprint arXiv:1704.04503*, 2017. 2, 8

[2] C. Desai, D. Ramanan, and C. C. Fowlkes. Discriminative models for multi-class object layout. *IJCV*, 95(1):1–12, 2011. 3

[3] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, June 2010. 4

[4] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017. 8

[5] R. Girshick. Fast r-cnn. In *ICCV*, pages 1440–1448, 2015. 1, 2, 7

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014. 1, 2

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, 2017. 5, 6, 7, 8

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 2, 5, 6, 7

[9] P. Henderson and V. Ferrari. End-to-end training of object class detectors for mean average precision. In *ACCV*, pages 198–213. Springer, 2016. 3

[10] J. Hosang, R. Benenson, and B. Schiele. Learning non-maximum suppression. *arXiv preprint arXiv:1705.02950*, 2017. 2, 3

[11] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. *arXiv preprint arXiv:1611.10012*, 2016. 6

[12] T. Kong, A. Yao, Y. Chen, and F. Sun. Hypernet: Towards accurate region proposal generation and joint object detection. In *CVPR*, pages 845–853, 2016. 2

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012. 5

[14] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. Fully convolutional instance-aware semantic segmentation. *arXiv preprint arXiv:1611.07709*, 2016. 6

[15] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 2, 5, 6, 7, 8

[16] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 1, 2, 5, 8

[17] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 1, 4, 5

[18] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37. Springer, 2016. 1, 2, 8

[19] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015. 2, 6

[20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, 2016. 1, 2

[21] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016. 1

[22] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015. 1, 2, 3, 6, 7

[23] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel. Self-critical sequence training for image captioning. *arXiv preprint arXiv:1612.00563*, 2016. 5

[24] M. A. Sadeghi and A. Farhadi. Recognition using visual phrases. In *CVPR*, pages 1745–1752. IEEE, 2011. 3

[25] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, pages 761–769, 2016. 2, 8

[26] A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta. Beyond skip connections: Top-down modulation for object detection. *arXiv preprint arXiv:1612.06851*, 2016. 8

[27] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. 5

[28] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

[29] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014. 7

[30] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, pages 4278–4284, 2017. 6

[31] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992. 2, 4, 5

[32] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. *arXiv preprint arXiv:1611.05431*, 2016. 6