Improving Object Detection with Inverted Attention

Zeyi Huang
Carnegie Mellon University
zeyih@andrew.cmu.edu

Wei Ke
Carnegie Mellon University
weik@andrew.cmu.edu

Dong Huang
Carnegie Mellon University
donghuang@cmu.edu

Abstract

Improving object detectors against occlusion, blur and noise is a critical step to deploy detectors in real applications. Since it is not possible to exhaust all image defects through data collection, many researchers seek to generate hard samples in training. The generated hard samples are either images or feature maps with coarse patches dropped out in the spatial dimensions. Significant overheads are required in training the extra hard samples and/or estimating drop-out patches using extra network branches. In this paper, we improve object detectors using a highly efficient and fine-grain mechanism called Inverted Attention (IA). Different from the original detector network that only focuses on the dominant part of objects, the detector network with IA iteratively inverts attention on feature maps and puts more attention on complementary object parts, feature channels and even context. Our approach (1) operates along both the spatial and channels dimensions of the feature maps; (2) requires no extra training on hard samples, no extra network parameters for attention estimation, and no testing overheads. Experiments show that our approach consistently improved both two-stage and single-stage detectors on benchmark databases.

1. Introduction

Improving object detectors against image defects such as occlusion, blur and noise is a critical step to deploy detectors in real applications. Recent efforts by computer vision community have collected extensively training data on different scenes, object categories, shapes and appearance. However, it is yet not possible to exhaust all image defects captured under camera shake, dust, fade lighting and tough weather conditions. Moreover, deep learning approaches are highly biased by data distribution, while it is very difficult to collect data with uniform combinations of object features and defects.

Besides waiting for more data collection and annotation, a fundamental way to improve detectors is to improve the training approach. Four main attempts have been recently explored by the computer vision community: (1) Selecting subsets of training samples by hard example mining [12]. This approach does not generalize well to unseen images defects. (2) Penalizing the occluded bounding boxes by occlusion-aware losses [17, 21]. These approaches do not generalize well to unseen occlusion. (3) Synthesizing image defects by hard example generation [13, 16, 22, 23]. These approaches typically drop out big patches in spatial dimension and the new samples to be trained increase exponentially. (4) Highlighting desired fine-grain features by attention estimation [3, 6, 11, 18, 24]. Estimating the attention masks usually require extra networks and therefore increase the training overheads.

In this paper, we improve object detector using a highly efficient and fine-grain mechanism called Inverted Attention (IA). IA is implemented as a simple module added to

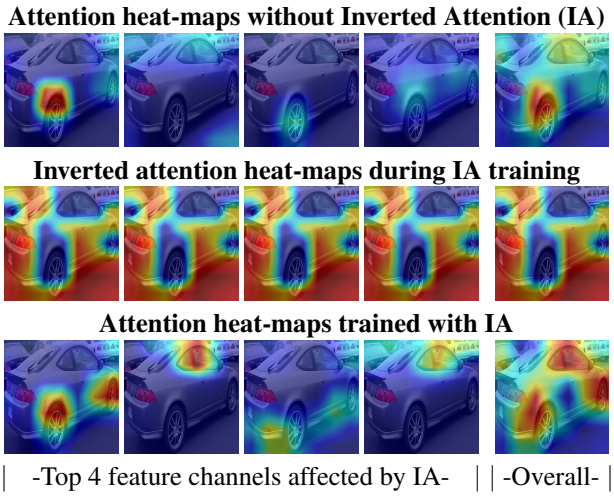


Figure 1: Attention heat-maps visualized at the ROI feature map ($[7 \times 7 \times 512]$) of VGG16 Fast-RCNN. Each $[7 \times 7]$ attention heatmap is linearly interpolated and superposed on the $[224 \times 224]$ object ROI image patch. The first 4 columns are respectively the attention heat-maps at the top 4 channels affected by IA. The last column, "Overall", denotes the overall attention summed over all 512 channels. (This figure is best viewed in color) 加入attention训练的更明显

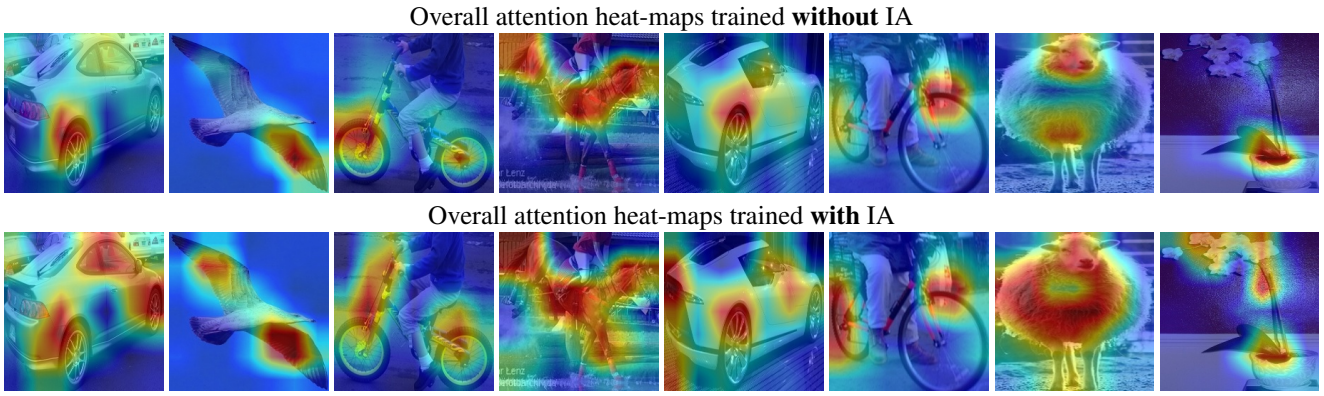


Figure 2: More examples of overall attention heat-maps trained without Inverted Attention (**Upper row**), and trained with Inverted Attention (**Lower row**).

the standard back-propagation operation. In every training iteration, IA computes the gradient of the feature maps produced at the feature extraction network (also called the backbone network) using object classification scores, and iteratively invert the attention of the network. Different from the original detector network that only focuses on the small parts of objects, the neural network with IA puts more attention on complementary spatial parts of the original network, feature channels and even the context. The IA module only changes the network weights in training and does not change any computation in inference.

Fig 1 (the upper row) visualizes the attention in a standard Fast-RCNN detector. The attention is visualized as heatmaps superposed on the object. The red pixels of the heat-maps denote high attention, while the blue pixels denote low attention. During our IA training, the original attention heat-maps are inverted as Fig 1 (the middle row) and proceed to the next iteration. After the IA training finishes, the network produces new attention heat-maps, see Fig 1 (the lower row). Observe that, the detector network trained with IA focuses on more comprehensive features of the objects, making it more robust to the potential defects of the individual pixels. Fig 2 shows more examples of overall attention produced without IA training (the upper row) and with IA training (the lower row). Comparing to the existing approaches, our approach (1) operates along both the spatial and channels dimensions of the feature maps; (2) requires no extra training on hard samples, no extra network parameters for attention estimation, and no testing overheads. We evaluated IA on both the two-stage and single-stage detectors on benchmark databases, and produced significant and consistent improvement.

2. Related Work

There are three kind of approaches to improve object detectors against noisy data.

2.1. Occlusion-Aware Loss

[21] proposed a aggregation loss for R-CNN based person detectors. This loss enforced proposals of the same object to be close. Instead of using a single RoI pooling layer for a person, the authors used a five-part pooling unit for each person to deal with occlusions. [17] proposed a new bounding box regression loss, termed repulsion loss. This loss encourages the attraction by target, and the repulsion by other surrounding objects. The repulsion term prevents the proposal from shifting to surrounding objects thus leading to more crowd-robust localization. [25] proposed two regression branches to improve pedestrian detection performance. The first branch is used to regress full body regions. The second branch is used to regress visible body regions.

2.2. Hard Sample Generation

Image based generation: [23] randomly occludes several rectangular patches of images during training. [7] occludes rectangular patches of image guided by the loss of the person Re-Identification task. [13] improves weakly-supervised object localization by randomly occluding patches in training images. [16] learns an adversarial network that generates examples with occlusions and deformations. The goal of the adversary is to generate examples that are difficult for the object detector to classify.

Feature based generation: [5] selects weighted channel to dropout for the regularization of CNNs. [16] uses predicted occlusion spatial mask to dropout for generating hard positive samples. [22] re-weights feature maps according to three kinds of channel-wise attention mechanism. [14] adaptively dropout input features to obtain various appearance changes using random generated masks in video tracking. Our IA approach conducts both channel-wise and spatial-wise dropout on features.

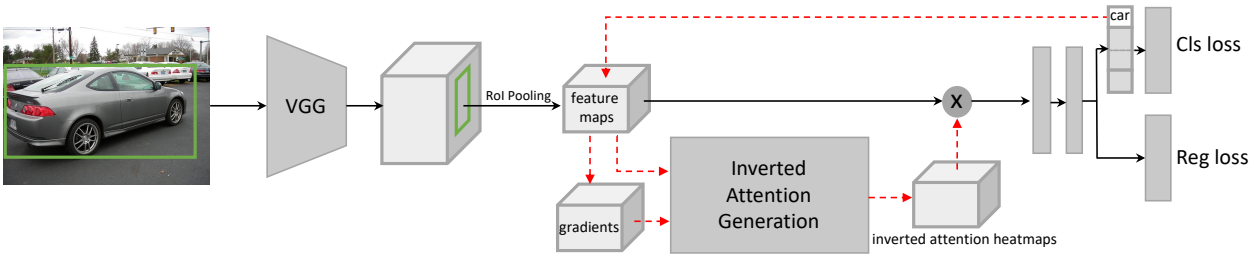


Figure 3: Architecture of Inverted Attention Network (IAN) based on a RNN object detection network. Light gray blocks denote tensors. Dark gray blocks denote operations. The data flows along the red dashed arrows are only needed in training. The inference only require the flows along the black arrows. 只在训练中用attention

2.3. Attention Estimation

Recent methods incorporate attention mechanism to improve the performance of CNNs. [3, 6, 18] integrate channel-wise or spatial-wise attention network branches to the feed-forward convolutional neural networks. The estimated attention maps are multiplied to the original feature map for various CNN tasks. All these methods introduce extra network branches to estimate the attention maps. [19] improves the performance of a student CNN network by transferring the attention maps of a powerful teacher network. Deconvnet [20] and guided-backpropagation [15] improve gradient-based attention using different backpropagation strategy. CAM [24] converts the linear classification layer into a convolutional layer to produce attention maps for each class. Grad-CAM [11] improve CAM and is applicable to a wide variety of CNN model.

In our method, we compute inverted attention guided by gradient-based attention and Grad-CAM. Our network does not require extra network parameters or teacher networks.

3. Inverted Attention for Object Detection

We take the R-CNN [4] based framework to illustrate our Inverted Attention Network (IAN) (Fig. 3). An Inverted Attention Generation Module (Fig. 4) is added to the R-CNN detection network, and operates on the ROI featrue maps. Note that this module consists of only few simple operations such as pooling, threshold and element-wise product. No extra parameter is needed to learn. In the rest of the paper, we show how this simple change can effectively improve the original network to overcome image defects.

3.1. Inverted Attention Generation Module

The attention mechanism identifies the most representative receptive field of an object. Highlighting the object with attention heatmaps encourages discrimination between object classes, meanwhile, decreases the diversity of features within the same class. However, the diversity of features is the key to generalize an object detector to unseen object instances and image defects. The proposed Inverted Attention

att可以鉴定最优代表性的感受野，提升判别性，但会减少特征的多样性，这是泛化性的保证

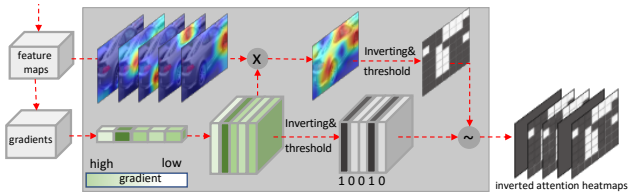


Figure 4: Detailed architecture of the Inverted Attention Generation Module in Fig. 3. This module consists of only simple operations such as pooling, threshold and element-wise product. No extra parameter is needed to learn. This module only operates during training, not in testing.

training approach aims to alleviate the conflict and find the optimal trade-off between discrimination and diversity.

避免冲突，找到判别性与多样性的最佳平衡

As shown in Fig. 4, the Inverted Attention Generation Module consists of two simple operations: (1) Gradient-Guided Attention Generation: computing the the gradients at feature maps, by back-warding only the classification score on the ground-truth category, (2) Attention Inversion: reversing element values of the attention tensor to produce IA heat-maps.

Gradient-Guided Attention Generation: In training phase of the convolutional neural networks, gradients of feature maps in the back-propagation operation encode how sensitive the output prediction is with respect to changes at the same location of the feature maps. If small changes at an element of feature maps have a strong effect on the network outputs, then the network will be updated to pay more attention on that feature element. With this principle, we use the gradient to generate attention map in our approach (see details in Fig.4).

Denote the gradient tensor as G and feature tensor as F. Both of them are of size $H \times W \times C$, where H, W, C are the height, width and channel number of G and F, respectively. A global pooling is applied on the gradient tensor to produce a weight vector W with size $C \times 1$. We compute a

全局池化得到C维梯度

(a) Epoch 1 (b) Epoch 2 (c) Epoch 3 (d) Epoch 4 (e) Epoch 5 (f) Epoch 6 (g) Epoch 7 (h) Epoch 8

Figure 5: Network attention evolves at different epochs during the IA training. Each sub-image shows the 2D attention heat-maps superposed on object ROI. The attention maps are visualized at the ROI feature map ($[7 \times 7 \times 512]$) of the VGG16 backbone network in Fast-RCNN.

gradient guided attention map following [11],

$$M = \sum_{i}^{C} w_i * F^{(i)}, \tag{1}$$

where w_i is the i-th element of W, and $F^{(i)}$ is the i-th channel map of F. As shown in Fig. 4, the high values in gradient correspond to the receptive field of and trunk in the car sample, while small values correspond to the receptive field of door of the car and background.

Attention Inversion: In the standard training process, the gradient descent algorithm forces the attention map to converge to a few most sensitive parts of objects, while ignores the other less sensitive parts of objects. The IA training conducts iterative inverting of the original attention tensor as the *Inverted Attention* tensor, which forces the network to detect object based on their less sensitive parts. Specifically, we generate a spatial-wise inverted attention map and a channel-wise inverted attention vector, and then combine them to produce the final attention maps.

强调背景不敏感部分

The spatial-wise inverted attention map $A^s = \{a_i^s\}$ is computed as

$$a_i^s = \begin{cases} 0 & \text{if } m_i > T_s \\ 1 & \text{else} \end{cases}, \tag{2}$$

where a_i^s and m_i are the elements of A^s and M at the i-th pixel, respectively. T_s is the threshold for spatial-wise attention map. From Eq. 2, spatial-wise inverted attention map pays more attention to the area of the sample with small gradient value.

Observe that the weight vector W serves as a sensitivity measure for channels of feature maps. A threshold T_c is used to compute the channel-wise inverted attention vector $A^c = \{a_j^c\}$,

$$a_j^c = \begin{cases} 0 & \text{if } w_j > T_c \\ 1 & \text{else} \end{cases}. \tag{3}$$

The final Inverted attention map $A = \{a_{i,j}\}$ is computed as

$$a_{i,j} = \begin{cases} a_i^s & \text{if } a_j^c = 0 \\ 1 & \text{else} \end{cases}. \tag{4}$$

Fig. 5) illustrates how network attention evolves at different epochs during the IA training. The IA training iteratively guides the neural network to extract features on the whole object sample.

3.2. Inverted Attention Network

As shown in Fig. 3, the Inverted Attention Network (IAN) is basically built by adding Inverted Attention Generation Module (Fig. 4) to the R-CNN based detection network, and operates on the ROI feature maps.

Given an input image, the backbone of the R-CNN framework, *i.e.*, VGG or ResNet, takes the whole image as an input and produces feature maps. The region proposals are generated from these feature maps by region proposal network (RPN) or pre-computed region proposal candidates. Then the RoI-pooling layer generates a fixed size feature maps for each object proposal. These feature maps after RoI pooling then go through fully connected layers for object classification and bounding box regression.

The R-CNN can be trained end-to-end by optimizing the following two loss functions:

$$L_{rpn} = L_{cross-entropy} + L_{rpn\_reg}, \tag{5}$$

$$L_{rcnn} = L_{softmax} + L_{rcnn\_reg}, \tag{6}$$

where $L_{cross-entropy}$ and L_{rpn_reg} are the cross-entropy loss and L1 loss for RPN network. $L_{cross-entropy}$ and L_{rpn_reg} are the softmax loss and L1 loss for RCNN network. $L_{rpn} + L_{rcnn}$ are jointly optimized in the Faster-RCNN framework, and L_{rcnn} is optimized in the Fast-RCNN framework.

In the backward stage, the gradient is computed by back-propagating the classification loss only on the ground-truth category, which is used for inverted attention generation module. With the generated Inverted Attention map, an element-wise product layer between feature maps and IA heat-maps is used for feature refinement, as

$$F_{new} = F. * A, \tag{7}$$

where $.*$ indicates element-wise multiplication. The refinement is conducted at element-level, i.e., along both the spatial and channels dimensions of the feature maps.

After these operations, the refined features are forwarded to compute the detection loss, and then the loss is back-propagated to update the original network parameters. The training process of IAN is summarized in Algorithm 1.

Algorithm 1 Training Process of IAN

Input: The images with ground-truth x_i, y_i $(i = 1, \cdots, n)$.

Output: Object detection model.

1: **for** each iteration **do**
2: Generating region proposal by the RPN network;
3: Getting the the feature map of the region proposal by ROI pooling, as F shown in Fig. 3;
4: Computing gradient G by back-warding the classification loss on the ground-truth category;
5: Computing the gradient-guided attention map with Eq. 1;
6: Achieving spatial-wise and channel-wise inverted attention maps with Eq. 2 and Eq. 3;
7: Refining feature map F with inverted attention map with Eq. 7;
8: Computing RPN loss and classification loss with Eq. 5 and Eq. 6;
9: Back-propagation.
10: **end for**

3.3. Discussion

The high attention regions learned by the original network represent the most common features shared by the training samples. These features are discriminative enough on the training data while may not be enough for the testing data, especially when high attention regions are corrupted by the unseen image defects.

Most top improvements on original networks were reached by discovering more discriminative features. For instance, Image based Hide-and-Seek (HaS) [23] and feature based A-Fast-RCNN [16]. HaS randomly hides patches in a training image, forcing the network to seek discriminative features on remaining patches of the images. A-Fast-RCNN finds the best patches to occlude by estimating a occlusion mask with a generation and adversary network.

Our IA approach fuses the advantages of both approaches in the training steps of the original detector network. The new discriminative features are iteratively discovered (see Fig. 5) by inverting the original attention. IA finds discriminative features in all object parts, feature channels and even context. This process requires no extra training epochs on hard samples and no extra network parameters to estimate occlusion mask.

Note that, IAN is not limited to two-stage detectors like Fast-RCNN and Faster R-CNN. We also tried IAN on single-stage detectors such as Single Shot MultiBox Detector(SSD) in our experiments section 4.2.2.

4. Experiments

Inverted Attention Network (IAN) was evaluated on three widely used benchmarks: the PASCAL VOC2007, PASCAL VOC2012 [2], and MS-COCO [8] datasets. In the following section, we first introduce the experimental settings, then analyze the effect of the Inverted Attention module. Finally, we report the performance of IAN and compare it with the state-of-the-art approaches.

We used Faster-RCNN, Fast-RCNN and SSD object detectors as our baselines. Our IANs are simply constructed by adding the IA module to each baseline network. VGG16 and ResNet-101 were used as the backbone feature extractors. By default, Fast R-CNN with VGG16 were used in ablation study. The standard Mean Average Precision (mAP) [1] are used as the evaluation metric. For PASCAL VOC, we report mAP scores using IoU thresholds at 0.5. For the COCO database, we use the standard COCO AP metrics.

4.1. Experimental Settings

PASCAL VOC2007: All models were trained on the VOC2007 trainval set and the VOC2012 trainval set, and tested on the VOC2007 test set. For Fast-RCNN, we followed the training strategies in [16]: set the learn rate to $2e^{-3}$ for the first 6 epochs, and decay it to $2e^{-4}$ for another 2 epochs. We used batch size 2 in training, and used VGG16 as the backbone networks for all the ablation study experiments on the PASCAL VOC dataset. For Faste-RCNN, we followed the training strategies in [10]: set the learn rate to $2e^{-3}$ for the first 6 epochs, and decay it to $2e^{-4}$ for another 2 epochs. We used the batch size 2 in training. We also report the results of ResNet101 backbone on these models. For SSD, we followed the training strategies in [9]: set the learn rate to $1e^{-3}$ for the first 6 epochs, and decay it to $1e^{-4}$ and $1e^{-5}$ for another 2 and 1 epochs. We use the default batch size 32 in training.

PASCAL VOC2012: All models were trained on the VOC2007 trainval set, VOC2012 trainval set and VOC2007 test set, and then tested on the VOC2012 test set. For Faster-RCNN and SSD experiments, we follow the exact same training strategies as the VOC2007 training above.

COCO: Following the standard COCO protocol, training and evaluation were performed on the $120k$ images in the trainval set and the $20k$ images in the test-dev set respectively. For Faster-RCNN, we set the learn rate to $1e^{-2}$ for the first 4 epochs, decay it to $1e^{-3}$ for another 4 epochs and $1e^{-4}$ for the last 2 epochs. We used batch size 16 in training, and used ResNet101 as the backbone networks. For SSD, we set the learn rate to $1e^{-3}$ for the first 6 epochs, and decay it to $1e^{-4}$ and $1e^{-5}$ for another 2 and 1 epochs. We use the default batch size 32 in training.

4.2. Implement Details

In the two-stage detection framework, given the feature maps after the ROI pooling layer, we reweight the feature maps guided by inverted attention. For spatial-wise inverted attention map $H \times W$, we dropout top 33% pixels with highest values. For channel-wise inverted attention map, we dropout top 80% pixels with highest values. In the single-stage detection framework, given several feature maps with different size, we follow the same strategies in above two-stage framework to reweight the feature maps. In order to prevent netwrok from overfitting to the reweighted feature maps, we only reweight 20% feature maps, and leave the rest 80% feature maps unchanged.

4.3. Results on PASCAL VOC2007

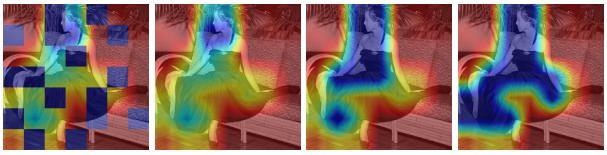
To verify the effectiveness of Inverted Attention, we first conducted ablation study on two key factors of IA, *i.e.*, inversion strategies and inversion orientation. By taking the best settings in the ablation study, we then show the results compared with baselines and state-of-the-art.

4.3.1 Ablation Study

The following ablation study is conducted on the Fast-RCNN with the VGG16 backbone.

Inversion Strategies: Four inversion strategies were evaluated: Random inversion, Overturn inversion, Hard-threshold inversion, and Soft-threshold inversion. Table 1 shows that all the four inverting strategies improve the performance of the baseline. Fig. 6 visualize the four strategies on the same on the same object.

As shown in Fig. 6a, by randomly selecting pixels on convolutional feature maps and setting them to 0, the mAP improves from 69.1% to 70.3%. However, random inversion loses the contextual information, meaning that even in the same semantic part, some pixels were kept while others were discarded. As shown in Fig. 6b Overturn inversion achieves inverted attention map by IA = 1.0 - A, which increases the weight of background and suppresses the foreground. Overturn inversion gets the mAP of 70.6%, which is 1.5 percentages better than baseline. To keeping the weights of background, we take two thresholding methods to suppresses all pixels in attention map which are large than the threshold. The hard-threshold is shown in Fig. 6c, which takes 0.5 as threshold. While the soft-threshold adopts a sorting algorithm and suppresses the top 33% pixels. Hard-threshold achieves 70.9% mAP and soft-threshold achieves 71.4%, which are 1.8% and 2.3% better than the baseline, respectively.



(a) (b) (c) (d)

Figure 6: Visualization of four inversion strategies. From (a) to (d), it illustrates inverted attention map by random, overturn, hard-threshold, and soft-threshold, respectively.

Method	mAP
Baseline (Fast-RCNN + VGG16)	69.1
Random	70.3
Overturn	70.6
Hard-threshold	70.9
Soft-threshold	**71.4**

Table 1: Ablation study on inversion strategies.

Method	mAP
Spatial	71.4
Channel	70.9
Spatial + Channel	**71.6**

Table 2: Ablation study on inverted orientations.

	Method	mAP
ROI Feature IA	Fast-RCNN	69.1
	Fast-RCNN + IA (ours)	**71.4**
Full Feature IA	Fast-RCNN	69.1
	Fast-RCNN + IA (ours)	**70.2**
	SSD300	77.2
	SSD300 + IA (ours)	**77.8**

Table 3: Full feature IA v.s. ROI feature IA on VOC2007.

Inversion Orientation: Using the soft-threshold inversion strategy, we further studied two inversion orientations in Table 2. The spatial inversion attention conducts inversion over all channels, while the channel inversion attention conducts inversion only on a subset of channels. Conducting spatial or channel inversion produced 71.4% and 70.9%, respectively. Conducting both the spatial and channel inver-

Method	Train	Backbone	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	bike	person	plant	sheep	sofa	train	TV
FRCN [4]	07	VGG16	69.1	**75.4**	80.8	67.3	59.9	37.6	**81.9**	80.0	84.5	50.0	77.1	68.2	81.0	82.5	74.3	69.9	28.4	71.1	70.2	75.8	66.6
ORE [23]	07	VGG16	71.0	75.1	79.8	69.7	**60.8**	46.0	80.4	79.0	83.8	51.6	76.2	67.8	81.2	**83.7**	76.8	73.8	**43.1**	70.8	67.4	78.3	**75.6**
Fast+ASTN [16]	07	VGG16	71.0	74.4	81.3	67.6	57.0	46.6	81.0	79.3	**86.0**	**52.9**	75.9	**73.7**	**82.6**	83.2	**77.7**	72.7	37.4	66.3	**71.2**	78.2	74.3
Fast+IA(ours)	07	VGG16	**71.6**	74.9	**82.0**	**71.8**	59.1	**47.6**	80.9	**80.5**	85.2	51.2	**77.2**	71.6	81.3	83.6	77.0	**74.1**	39.3	**71.1**	70.0	**79.2**	74.0
FRCN [4]	07	ResNet101	71.8	**78.7**	82.2	71.8	55.1	41.7	79.5	80.8	88.5	53.4	81.8	72.1	87.6	85.2	80.0	72.0	35.5	71.6	75.8	**78.3**	64.3
Fast+ASTN [16]	07	ResNet101	73.6	75.4	**83.8**	75.1	61.3	44.8	**81.9**	**81.1**	87.9	57.9	81.2	**72.5**	87.6	**85.2**	80.3	74.7	**44.3**	72.2	76.7	76.9	71.4
Fast+IA(ours)	07	ResNet101	**74.7**	77.3	81.2	**78.1**	**62.6**	**52.5**	77.8	80.0	**88.7**	**58.6**	**81.8**	71.4	**87.9**	84.2	**81.4**	**76.6**	44.0	**77.1**	**79.1**	76.9	**77.2**
Faster [10]	07	VGG16	69.9	70.0	**80.6**	**70.1**	**57.3**	49.9	78.2	80.4	82.0	**52.2**	75.3	**67.2**	80.3	79.8	75.0	76.3	39.1	68.3	67.3	**81.1**	67.6
Faster+IA(ours)	07	VGG16	**71.1**	**73.4**	78.5	68.3	54.7	**56.1**	**81.0**	**85.5**	**84.3**	48.4	**77.9**	61.7	**80.5**	**82.6**	**75.3**	**77.5**	**47.0**	**71.7**	**68.8**	76.0	**72.5**
Faster	07	ResNet101	75.1	76.5	79.7	77.7	66.4	61.0	83.3	86.3	**87.5**	53.6	81.1	66.9	85.3	**85.1**	77.4	**78.9**	**50.0**	74.1	**75.8**	78.9	75.4
Faster+IA(ours)	07	ResNet101	**76.5**	**77.9**	**82.9**	**78.4**	**67.2**	**62.2**	**84.2**	**86.9**	87.2	**55.5**	**85.6**	**69.1**	**87.0**	85.0	**81.4**	78.8	48.4	**79.4**	75.0	**83.2**	**75.4**
Faster [10]	07+12	VGG16	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	**81.9**	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
Faster+IA(ours)	07+12	VGG16	**76.8**	**78.5**	**81.1**	**76.8**	**67.2**	**63.9**	**87.1**	**87.7**	**87.8**	**59.3**	81.1	**72.9**	**84.8**	**86.7**	**80.5**	**78.7**	**50.9**	**76.9**	**74.2**	**83.1**	**76.5**
Faster [10]	07+12	ResNet101	76.4	79.8	80.7	76.2	68.3	55.9	85.1	85.3	**89.8**	56.7	**87.8**	69.4	**88.3**	**88.9**	80.9	78.4	41.7	78.6	79.8	85.3	72.0
Faster+IA(ours)	07+12	ResNet101	**81.1**	**85.3**	**86.8**	**79.7**	**74.6**	**69.4**	**88.4**	**88.7**	88.8	**64.8**	87.3	**74.7**	87.7	88.6	**85.3**	**83.5**	**53.9**	**82.7**	**81.5**	**87.8**	**80.9**

Table 4: Object detection Average Precision (AP) tested on VOC2007.

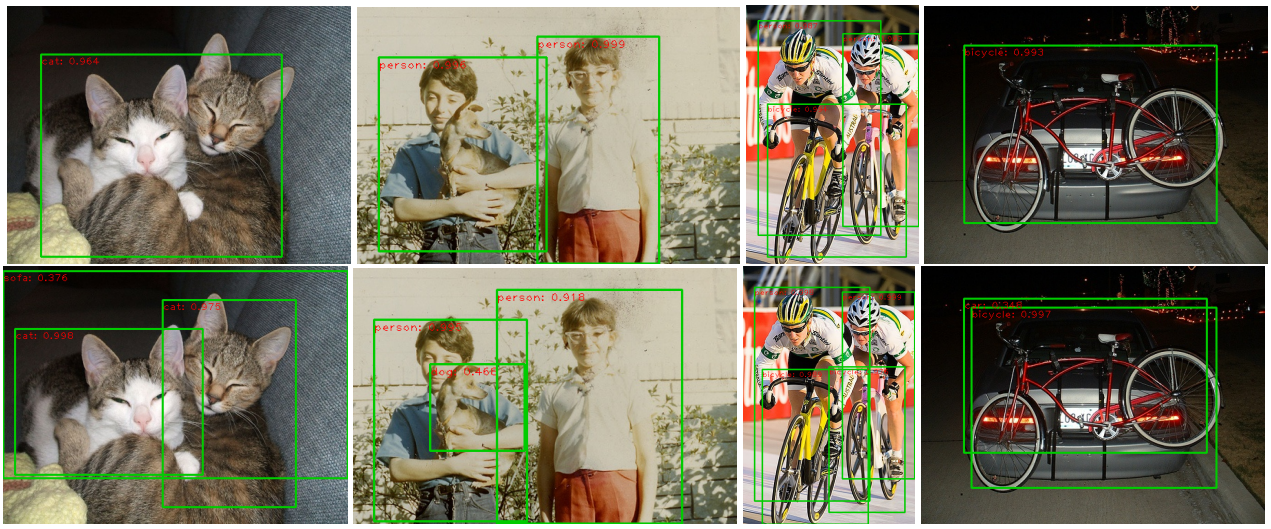


Figure 7: Object detection examples in VOC2007 with ResNet101-Faster-RCNN (top) and its IAN version (bottom).

sion, the performance is further improved to 71.6%.

4.3.2 Full Feature IA v.s. ROI Feature IA

Inverted Attention is a plugin module attached to certain feature maps of neural networks. We illustrate the choices of feature maps to conduct IA on the two-stage object detection framework, e.g., Fast-RCNN and Faster R-CNN, and the single-stage object detection framework, *e.g.*, Single Shot MultiBox Detector(SSD).

In Fast-RNN, IA can be plugged on the feature maps either before, called Full Feature IA, or after ROI pooling, called ROI Feature IA. In Full Feature IA, The features of whole image is refined, while in ROI Feature IA, the features of different ROIs are refined separately. The performance comparison is shown in Table 3. IA improves the baselines no matter which strategy was used. ROI Feature IA performs better than the Full Feature IA. We believe this is because that ROI Feature IA confined the inverted attention in the vicinity of objects and is more likely to learn object-related features. Whereas for the Full Feature IA, the inverted attention covers mostly the background, making it difficult to learn object-related features.

As there is no ROI pooling in SSD, the IA can only be plugged on the full image feature. The mAP of SSD300 was increased from 77.2% to 77.8%.

4.3.3 Comparing with Baselines and State-of-the-Art

We first present extensive performance comparison on the PASCAL VOC 2007 with Fast-RCNN (denoted as "FRCN"), Faster-RCNN (denoted as "Faster") and the state-of-the-art hard-sample generation approaches ORE [23] and Fast-RCNN with ASTN [16] (denoted as "Fast+ASTN"). These approaches only provided results on Fast-RCNN. The results are compared in Table 4.

Method	Train	Backbone	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	bike	person	plant	sheep	sofa	train	TV
Faster [10]	07++12	ResNet101	73.8	86.5	81.6	77.2	58.0	51.0	78.6	76.6	93.2	48.6	80.4	59.0	**92.1**	85.3	84.8	80.7	48.1	77.3	66.5	84.7	65.6
Faster+IA(ours)	07++12	ResNet101	**79.2**	**87.7**	**86.7**	**80.3**	**68.1**	**62.1**	**81.0**	**84.7**	**93.8**	**61.8**	**84.2**	**63.1**	92.0	**87.4**	**86.6**	**85.8**	**61.0**	**84.6**	**72.4**	**86.5**	**73.8**
SSD300 [9]	07++12	VGG16	75.8	**88.1**	82.9	74.4	61.9	47.6	**82.7**	78.8	91.5	58.1	80.0	64.1	89.4	**85.7**	85.5	82.6	50.2	79.8	**73.6**	**86.6**	**72.1**
SSD300+IA(ours)	07++12	VGG16	**77.9**	87.5	**85.0**	**79.1**	**66.6**	**60.4**	80.0	**83.6**	**92.3**	**59.8**	**82.3**	**64.8**	**89.9**	85.6	**85.7**	**84.5**	**59.5**	**82.2**	71.8	85.8	71.6

Table 5: Object detection Average Precision (AP) tested on VOC2012.

Method	Train	Backbone	$AP_{[0.5,0.95]}$	$AP_{0.5}$	$AP_{0.75}$	AP_s	AP_m	AP_l
Faster [10]	trainval	VGG16	21.9	42.7	23.0	6.7	25.2	36.4
Faster+++ [10]	trainval35k	ResNet101	34.9	55.7	37.4	**15.6**	38.7	50.9
Faster+IA(ours)	trainval35k	ResNet101	**35.5**	**56.1**	**38.2**	14.9	**38.8**	**51.7**
SSD512 [10]	trainval35k	VGG16	28.8	48.5	30.3	10.9	31.8	**43.5**
SSD512+IA(ours)	trainval35k	VGG16	**29.6**	**49.8**	**31.4**	**11.9**	**32.4**	42.8

Table 6: Object detection Average Precision (AP) tested on COCO test-dev 2017.

With the VGG16 backbone and the VOC2007 training data, IA improved Fast-RCNN from 69.1% to 71.6%, and improves Faster-RCNN form 69.9% to 71.1%, which are 2.5% and 1.2% improvement respectively. With more powerful backbone, *i.e.*, ResNet101, Fast-RCNN and Faster-RCNN achieves better object detection performance than VGG16. By adding IA to them, the performance were consistently improved: for Fast-RCNN from 71.8% to 71.4%, and for Faster-RCNN from 75.1% to 76.5%, respectively. Using the training data from both VOC2007 and VOC2012, the mAPs of our approach were further improved to 76.8% with VGG16, and 81.1% with ResNet101.

Fig. 7 shows some detection examples from the VOC2007 test set using ResNet101 Faster-RCNN and its IAN version. These examples illustrate that IAN improves object detection to handle images defects such as heavy occlusions, faded pictures and shadows.

4.4. Results on PASCAL VOC2012 and COCO2017

For the PASCAL VOC2012 and COCO2017 datasets, we used Faster-RCNN to construct Inverted Attention Network. For both of the datasets, the evaluation results were produced by the official evaluation server.

The detection performance of PASCAL VOC2012 is shown in Table 5. For ResNet101 Faster-RCNN, IA increased mAP from 73.8% of the baseline to 79.2%, which is 5.4% improvement. The AP on 19 categories achieved consistent performance gain. For ResNet101 SSD300, IA increased mAP from 75.8% of baseline to 77.9%, which is 2.1% improvement. The AP on 14 categories achieved consistent performance gain. This demonstrates that our IAN can discover more discriminative features for a large variety of object classes.

The detection performance of COCO2017 is shown in Table 6. We used the official evaluation metrics for COCO. Faster-RCNN baseline with VGG16 trained on the train and validation set of COCO produced 21.9% for $AP_{[0.5,0.95]}$. Using the more powerful backbone ResNet101, $AP_{[0.5,0.95]}$ reached 35.5% even when the training data were reduced to the train and val35k subsets of COCO. SSD512 with VGG16 produced $AP_{[0.5,0.95]}$ 28.8. We plugged the IA module into Faster-RCNN with ResNet101 and SSD512 with VGG16 and consistently improved $AP_{[0.5,0.95]}$ $AP_{0.5}$ and $AP_{0.75}$ over the baselines.

The COCO evaluation server also gave the detection performance on small (AP_s), medium(AP_m), and large objects(AP_l). It is interesting to note that, in Table 6, IAN tends to boost the performance of the medium and large objects for Faster-RCNN where IA was conducted on ROI features. While for SSD512, when IA was conducted on full features, IAN favors small and medium objects.

5. Conclusion

We present IA as a highly efficient training module to improve the object detection networks. IA computes attention using gradients of feature maps during training, and iteratively inverts attention along both spatial and channel dimension of the feature maps. The object detection network trained with IA spreads its attention to the whole objects. As a result, IA effectively improves diversity of features in training, and makes the network robust to image defects. It is very attractive to explore the best configurations of IA module on all other computer vision tasks such as image classification, instance segmentation and tracking.

References

[1] M. Everingham, L. J. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. 5

[2] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 5

[3] J. Fu, J. Liu, H. Tian, Z. Fang, and H. Lu. Dual attention network for scene segmentation. *arXiv preprint arXiv:1809.02983*, 2018. 1, 3

[4] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 3, 7

[5] S. Hou and Z. Wang. Weighted channel dropoutfor regularization of deep convolutional neural network. In *AAAI*, 2019. 2

[6] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 1, 3

[7] H. Huang, D. Li, Z. Zhang, X. Chen, and K. Huang. Adversarially occluded samples for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5098–5107, 2018. 2

[8] 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 *European conference on computer vision*, pages 740–755. Springer, 2014. 5

[9] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *Proceedings of the European conference on computer vision*, pages 21–37. Springer, 2016. 5, 8

[10] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 5, 7, 8

[11] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 618–626, 2017. 1, 3, 4

[12] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 761–769, 2016. 1

[13] K. K. Singh and Y. J. Lee. Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3544–3553, 2017. 1, 2

[14] Y. Song, C. Ma, X. Wu, L. Gong, L. Bao, W. Zuo, C. Shen, R. W. Lau, and M.-H. Yang. Vital: Visual tracking via adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8990–8999, 2018. 2

[15] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014. 3

[16] X. Wang, A. Shrivastava, and A. Gupta. A-fast-rcnn: Hard positive generation via adversary for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2606–2615, 2017. 1, 2, 5, 7

[17] X. Wang, T. Xiao, Y. Jiang, S. Shao, J. Sun, and C. Shen. Repulsion loss: detecting pedestrians in a crowd. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7774–7783, 2018. 1, 2

[18] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision*, pages 3–19, 2018. 1, 3

[19] S. Zagoruyko and N. Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016. 3

[20] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Proceedings of the European conference on computer vision*, pages 818–833, 2014. 3

[21] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li. Occlusion-aware r-cnn: detecting pedestrians in a crowd. In *Proceedings of the European Conference on Computer Vision*, pages 637–653, 2018. 1, 2

[22] S. Zhang, J. Yang, and B. Schiele. Occluded pedestrian detection through guided attention in cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6995–7003, 2018. 1, 2

[23] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang. Random erasing data augmentation. *arXiv preprint arXiv:1708.04896*, 2017. 1, 2, 5, 7

[24] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016. 1, 3

[25] C. Zhou and J. Yuan. Bi-box regression for pedestrian detection and occlusion estimation. In *Proceedings of the European Conference on Computer Vision*, pages 135–151, 2018. 2