

# Decoupled Classification Refinement: Hard False Positive Suppression for Object Detection

Bowen Cheng, Yunchao Wei, Honghui Shi, *Member, IEEE*, Rogerio Schmidt Feris, *Senior Member, IEEE*, Jinjun Xiong, *Member, IEEE*, and Thomas Huang, *Life Fellow, IEEE*

**Abstract**—In this paper, we analyze failure cases of state-of-the-art detectors and observe that most *hard false positives* result from classification instead of localization and they have a large negative impact on the performance of object detectors. We conjecture there are three factors: (1) Shared feature representation is not optimal due to the mismatched goals of feature learning for classification and localization; (2) multi-task learning helps, yet optimization of the multi-task loss may result in sub-optimal for individual tasks; (3) large receptive field for different scales leads to redundant context information for small objects. We demonstrate the potential of detector classification power by a simple, effective, and widely-applicable *Decoupled Classification Refinement* (DCR) network. In particular, DCR places a separate classification network in parallel with the localization network (base detector). With ROI Pooling placed on the early stage of the classification network, we enforce an adaptive receptive field in DCR. During training, DCR samples hard false positives from the base detector and trains a strong classifier to refine classification results. During testing, DCR refines all boxes from the base detector. Experiments show competitive results on PASCAL VOC and COCO without any bells and whistles. Our codes are available at: <https://github.com/bowenc0221/Decoupled-Classification-Refinement>.

**Index Terms**—Object Detection, Deep Learning.

## 1 INTRODUCTION

REGION-based approaches with convolutional neural networks (CNNs) [1]–[11] have achieved great success in object detection. Such detectors are usually built with separate classification and localization branches on top of shared feature extraction networks, and trained with multi-task loss. In particular, Faster RCNN [3] learns one of the first end-to-end two-stage detector with remarkable efficiency and accuracy. Many follow-up works, such as R-FCN [12], Feature Pyramid Networks (FPN) [13], Deformable ConvNets (DCN) [14], have been leading popular detection benchmark in PASCAL VOC [15] and COCO [16] datasets in terms of accuracy. Yet, few work has been proposed to study what is the full potential of the classification power in Faster RCNN styled detectors.

To answer this question, in this paper, we begin with investigating the key factors affecting the performance of Faster RCNN. As shown in Fig 1 (a), we conduct object detection on PASCAL VOC 2007 using Faster RCNN and count the number of false positive detections in different confidence score intervals (blue). Although only a small percentage of all false positives are predicted with high confidence scores, these samples lead to a significant performance drop in mean average precision (mAP). In particular, we perform an analysis of potential gains in mAP using Faster RCNN: As illustrated in Fig 1 (b), given the detection

results from Faster RCNN and a confidence score threshold, we assume that all false positives with predicted confidence score above that threshold were classified correctly and we report the correspondent hypothesized mAP. It is evident that by correcting all false positives, Faster RCNN could, hypothetically, have achieved 86.8% in mAP instead of 79.8%. Moreover, even if we only eliminate false positives with high confidences, as indicated in the red box, we can still improve the detection performance significantly by 3.0% mAP, which is a desired yet hard-to-obtain boost for modern object detection systems.

The above observation motivates our work to alleviate the burden of false positives and improve the classification power of Faster RCNN based detectors. By scrutinizing the false positives produced by Faster RCNN, we conjecture that such errors are mainly due to three reasons: (1) Shared feature representation for both classification and localization may not be optimal for region proposal classification, the mismatched goals in feature learning lead to the reduced classification power of Faster RCNN; (2) Multi-task learning in general helps to improve the performance of object detectors as shown in Fast RCNN [2] and Faster RCNN, but the joint optimization also leads to possible sub-optimal to balance the goals of multiple tasks and could not directly utilize the full potential on individual tasks; (3) Receptive fields in deep CNNs such as ResNet-101 [17] are large, the whole image are usually fully covered for any given region proposals. Such large receptive fields could lead to inferior classification capacity by introducing redundant context information for small objects.

Following the above argument, we propose a simple yet effective approach, named Decoupled Classification Refinement (DCR), to eliminate high-scored false positives and improve the region proposal classification results. DCR

- B. Cheng, Y. Wei and T. Huang are with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, 61801.  
E-mail: {bcheng9, yunchao, t-huang1}@illinois.edu
- H. Shi, R. Feris and J. Xiong are with IBM T.J. Watson Research Center, Yorktown Heights, NY, 10598.  
E-mail: Honghui.Shi@ibm.com, {rsferis, jinjun}@us.ibm.com

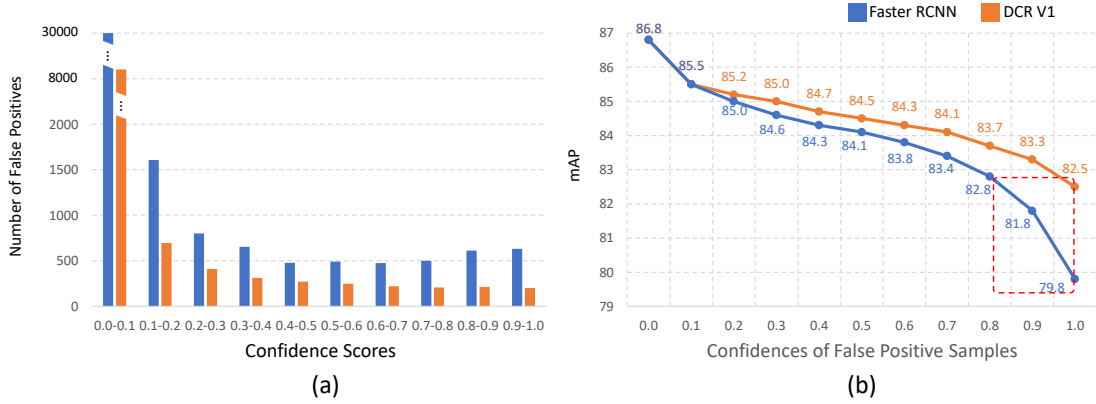


Fig. 1: (a) Comparison of the number of false positives in different ranges. (b) Comparison of the mAP gains by progressively removing false positives; from right to left, the detector is performing better as false positives are removed according to their confidence scores.

decouples the classification and localization tasks in Faster RCNN styled detectors. It takes input from a base detector, *e.g.* the Faster RCNN, and refine the classification results using a separate classification network which does not share features with the base detector. DCR samples *hard false positives*, namely the false positives with high confidence scores, from the base classifier, and then trains a stronger correctional classifier for the classification refinement. Designedly, we do not share any parameters between the Faster RCNN and our DCR module, so that the DCR module can not only utilize the multi-task learning improved results from region proposal networks (RPN) and bounding box regression tasks, but also better optimize the newly introduced module to address the challenging classification cases.

Experimental results show the benefit of decoupling classification and localization tasks in object detection and, more interestingly, we find a new speed-accuracy trade-off in object detection which is controlled by the amount of features shared between classification network and localization network. Namely, we find that the less features shared between classification and localization task, the better performance a detector will achieve and, in the mean while, the slower the detector is during inference time. We hope this new trade-off can motivate the society to find new directions of improving over current objection architectures.

We conduct extensive experiments based on different Faster RCNN styled detectors (*i.e.* Faster RCNN, Deformable ConvNets, FPN) and benchmarks (*i.e.* PASCAL VOC 2007 & 2012, COCO) to demonstrate the effectiveness of our proposed simple solution in enhancing the detection performance by alleviating hard false positives. As shown in Fig 1 (a), our approach can significantly reduce the number of hard false positives and boost the detection performance by 2.7% in mAP on PASCAL VOC 2007 over a strong baseline as indicated in Fig 1 (b). All of our experiment results demonstrate that our proposed DCR module can provide consistent improvements over various detection baselines, as shown in Fig 2 (a). Our contributions are threefold:

- 1) We analyze the error modes of region-based object detectors and formulate the hypotheses that might cause these failure cases.
- 2) We propose a set of design principles to improve the

classification power of Faster RCNN styled object detectors along with the DCR module based on the proposed design principles.

- 3) We are the first to show decoupling classification and localization helps object detection task and observe a new speed and accuracy trade-off in object detection.
- 4) Our DCR modules consistently bring significant performance boost to strong object detection systems on popular benchmarks and achieve competitive results. In particular, following common practice of using ResNet-101 as backbone without any bells and whistles, we achieve mAP of 84.2% and 81.2% on the classic PASCAL VOC 2007 and 2012 set, respectively, and 43.5% on the more challenging COCO *test-dev* set.

This paper is an extended version of our previous work [18]. In particular, we make three methodological improvements: the first one is that we propose a new DCR module that can be end-to-end trained together with the base object detector; the second one is that the proposed DCR module reduces the computation significantly without sacrificing accuracy by sharing some of the computation of regional feature at the image level; and the third one is that we come up with a inference technique that can further reduces the inference time while keeping a good speed-accuracy trade-off (Fig 2 (b)). In addition, we provide more results to demonstrate the effectiveness of our proposed method and we also perform more experiments to support our hypothesis. To justify decoupling features is important, we perform experiments to share different amount of features between classification and localization network (Table 2 (a)) and we observe that the less features shared the better the performance is (mAP increases from 78.4% to 83.0%). We further justify the importance of adaptive receptive field by placing ROI Pooling to different stages (Table 2 (b)) and we observe that the earlier ROI Pooling is placed the more adjustable receptive field can be and the better performance we get (mAP increases from 79.8% to 83.0%). These experiments also give new speed-accuracy trade-offs for object detectors.

The rest of the paper is organized as follows. In Section 2, some related works are introduced. In Section 3, we analyze

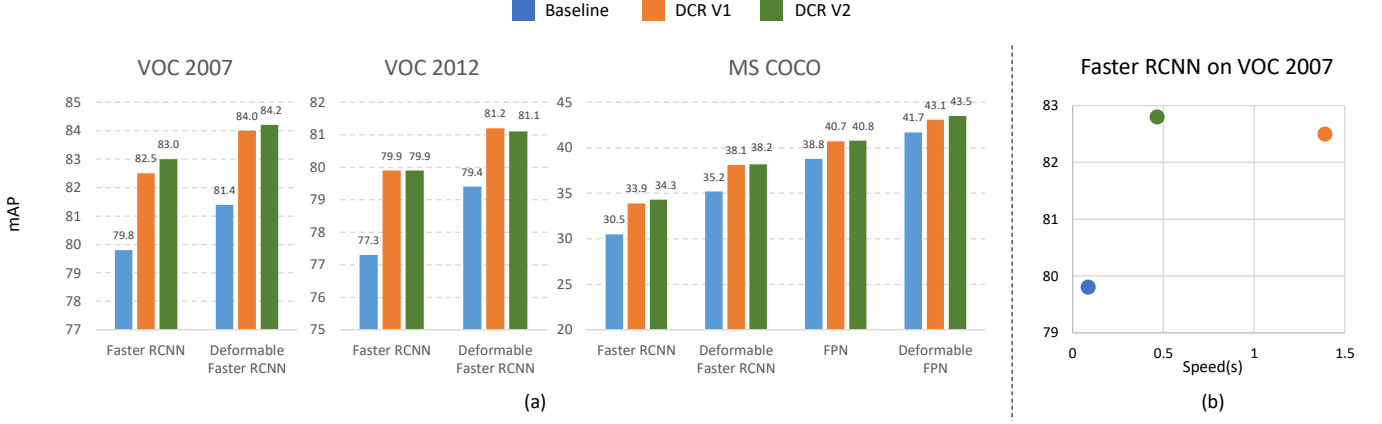


Fig. 2: (a) Comparison of our proposed DCR V1 and V2 with baseline in terms of different Faster RCNN series and benchmarks. (b) Speed-Accuracy comparison of DCR V1 and V2 upon Faster RCNN.

problems of current object detectors. In Section 4, the details of the proposed method are described. In Section 5, we extensively perform experiments and analysis. Finally, we conclude this paper in Section 7.

## 2 RELATED WORKS

### 2.1 Object Detection

Recent CNN based object detectors can generally be categorized into two-stage and single stage. One of the first two-stage detector is RCNN [1], where selective search [19] is used to generate a set of region proposals for object candidates, then a deep neural network to extract feature vector of each region followed by SVM classifiers. SPPNet [20] improves the efficiency of RCNN by sharing feature extraction stage and use spatial pyramid pooling to extract fixed length feature for each proposal. Fast RCNN [2] improves over SPPNet by introducing an differentiable ROI Pooling operation to train the network end-to-end. Faster RCNN [3] embeds the region proposal step into a Region Proposal Network (RPN) that further reduce the proposal generation time. R-FCN [12] proposed a position sensitive ROI Pooling (PSROI Pooling) that can share computation among classification branch and bounding box regression branch. Deformable ConvNets (DCN) [14] further add deformable convolutions and deformable ROI Pooling operations, that use learned offsets to adjust position of each sampling bin in naive convolutions and ROI Pooling, to Faster RCNN. Feature Pyramid Networks (FPN) [13] add a top-down path with lateral connections to build a pyramid of features with different resolutions and attach detection heads to each level of the feature pyramid for making prediction. Finer feature maps are more useful for detecting small objects and thus a significant boost in small object detection is observed with FPN. Most of the current state-of-the-art object detectors are two-stage detectors based of Faster RCNN, because two-stage object detectors produce more accurate results and are easier to optimize. However, two-stage detectors are slow in speed and require very large input sizes due to the ROI Pooling operation. Aimed at achieving real time object detectors, one-stage method, such as OverFeat [21], SSD [22], [23] and YOLO [24], [25], predict object classes and locations directly. Though single stage methods are much

faster than two-stage methods, their results are inferior and they need more extra data and extensive data augmentation to get better results. Our paper follows the method of two-stage detectors [1]–[3], but with a main focus on analyzing reasons why detectors make mistakes.

### 2.2 Classifier Cascade

The method of classifier cascade commonly trains a stage classifier using misclassified examples from a previous classifier. This has been used a lot for object detection in the past. The Viola Jones Algorithm [26] for face detection used a hard cascades by Adaboost [27], where a strong region classifier is built with cascade of many weak classifier focusing attentions on different features and if any of the weak classifier rejects the window, there will be no more process. Soft cascades [28] improved [26] built each weak classifier based on the output of all previous classifiers. Deformable Part Model (DPM) [29] used a cascade of parts method where a root filter on coarse feature covering the entire object is combined with some part filters on fine feature with greater localization accuracy. More recently, Li et al. [30] proposed the Convolutional Neural Network Cascade for fast face detection. Our paper proposed a method similar to the classifier cascade idea, however, they are different in the following aspects. The classifier cascade aims at producing an efficient classifier (mainly in speed) by cascade weak but fast classifiers and the weak classifiers are used to reject examples. In comparison, our method aims at improving the overall system accuracy, where exactly two strong classifiers are cascaded and they work together to make more accurate predictions. More recent Cascade RCNN [4] proposes training object detector in a cascade manner with gradually increased IoU threshold to assign ground truth labels to align the testing metric *i.e.* average mAP with IOU 0.5:0.05:0.95.

## 3 PROBLEMS WITH FASTER RCNN

Faster RCNN produces 3 typical types of hard false positives, as shown in Fig 3: (1) The classification is correct but the overlap between the predicted box and ground truth has low IoU, *e.g.*  $< 0.5$  in Fig 3 (a). This type of false negative boxes usually cover the most discriminative part



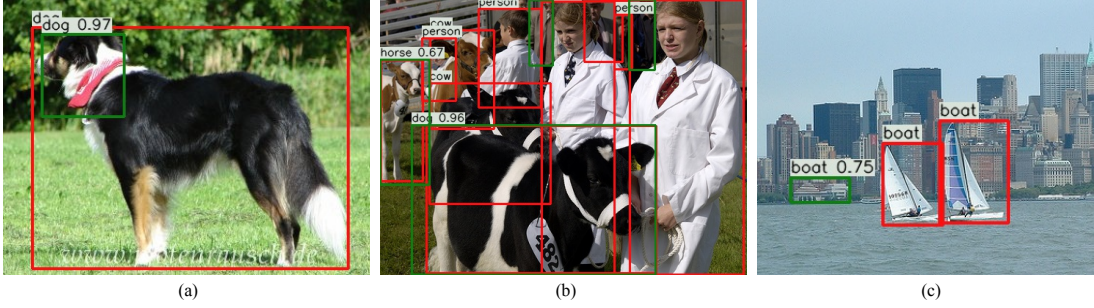


Fig. 3: Demonstration of hard false positives. Results are generated by Faster RCNN with 2 fully connected layers (2fc) as detector head [3], [13], red boxes are ground truth, green boxes are hard false positives with scores higher than 0.3; (a) boxes covering only part of objects with high confidences; (b) incorrect classification due to similar objects; (c) misclassified backgrounds.

and have enough information to predict the correct classes due to translation invariance. (2) Incorrect classification for predicted boxes but the IoU with ground truth are large enough, e.g. in Fig 3 (b). It happens mainly because some classes share similar discriminative parts and the predicted box does not align well with the true object and happens to cover only the discriminative parts of confusion. Another reason is that the classifier used in the detector is not strong enough to distinguish between two similar classes. (3) the detection is a “confident” background, meaning that there is no intersection or small intersection with ground truth box but classifier’s confidence score is large, e.g. in Fig 3 (c). Most of the background pattern in this case is similar to its predicted class and the classifier is too weak to distinguish. Another reason for this case is that the receptive field is fixed and it is too large for some box that it covers the actual object in its receptive field. In Fig 3 (c), the misclassified background is close to a ground truth box (the left boat), and the large receptive field (covers more than 1000 pixels in ResNet-101) might “see” too much object features to make the wrong prediction. Given above analysis, we can conclude that the hard false positives are mainly caused by the suboptimal classifier embedded in the detector. The reasons may be that: (1) feature sharing between classification and localization, (2) optimizing the sum of classification loss and localization loss, and (3) detector’s receptive field does not change according to the size of objects.

### 3.1 Problem with Feature Sharing

Detector backbones are usually adapted from image classification model and pre-trained on large image classification dataset. These backbones are originally designed to learn scale invariant features for classification. Scale invariance is achieved by adding sub-sampling layers, e.g. max pooling, and data augmentation, e.g. random crop. Detectors place a classification branch and localization branch on top of the same backbone, however, classification needs **translation invariant** feature whereas localization needs **translation covariant** feature. During fine-tuning, the localization branch will force the backbone to gradually learn translation covariant feature, which might potentially down-grade the performance of classifier.

### 3.2 Problem with Optimization

Faster RCNN series are built with a feature extractor as backbone and two task-specified branches for classifying

regions and the other for localizing correct locations. Denote loss functions for classification and localization as  $L_{cls}$  and  $L_{bbox}$ , respectively. Then, the optimization of Faster RCNN series is to address a Multi-Task Learning (MTL) problem by minimizing the sum of two loss functions:  $L_{detection} = L_{cls} + L_{bbox}$ . However, the optimization might converge to a compromising suboptimal of two tasks by simultaneously considering the sum of two losses, instead of each of them.

Originally, such a MTL manner is found to be effective and observed improvement over state-wise learning in Fast(er) RCNN works. However, MTL for object detection is not studied under the recent powerful classification backbones, e.g. ResNets. Concretely, we hypothesize that MTL may work well based on a weak backbone (e.g. AlexNet or VGG16). As the backbone is getting stronger, the powerful classification capacity within the backbone may not be fully exploited and MTL becomes the bottleneck.

### 3.3 Problem with Receptive Field

Deep convolutional neural networks have fixed receptive fields. For image classification, inputs are usually cropped and resized to have fixed sizes, e.g.  $224 \times 224$ , and network is designed to have a receptive field little larger than the input region. However, since contexts are cropped and objects with different scales are resized, the “effective receptive field” is covering the whole object.

Unlike image classification task where a single large object is in the center of an image, objects in detection task have various sizes over arbitrary locations. In Faster RCNN, the ROI pooling is introduced to crop object from 2-D convolutional feature maps to a 1-D fixed size representation for the following classification, which results in fixed receptive field (i.e. the network is attending to a fixed-size window of the input image). In such a case, objects have various sizes and the fixed receptive field will introduce different amount of context. For a small object, the context might be too large to focus on the object whereas for a large object, the receptive field might be too small that the network is looking at part of the object. Although some works introduce multi-scale features by aggregating features with different receptive fields, the number of sizes is still too small comparing with the number of various sizes of objects.

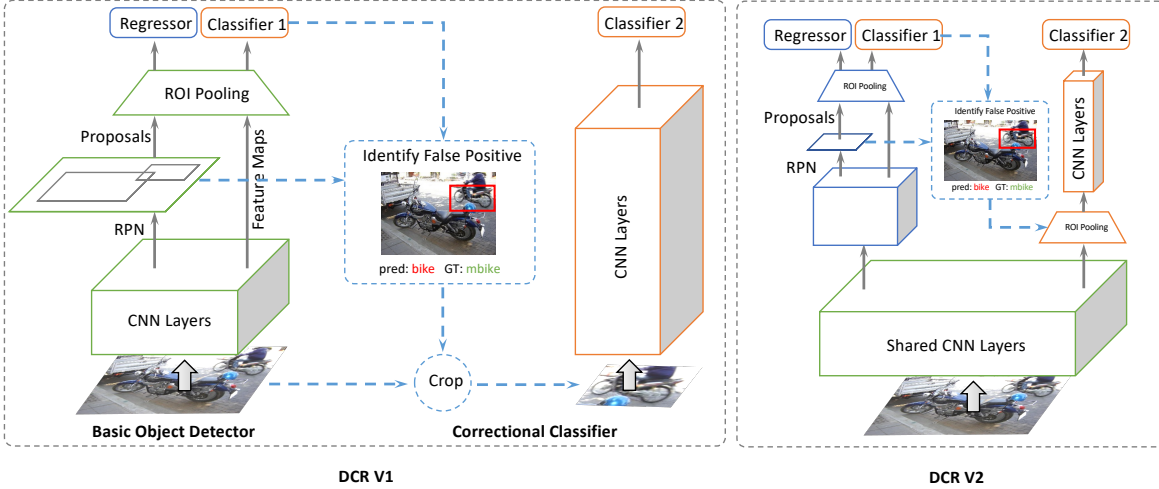


Fig. 4: Left: DCR V1 module [18]. Right: our proposed DCR V2 module.

## 4 DECOUPLED CLASSIFICATION REFINEMENT

In this section, we look back closely into the classic RCNN [1] method, and give an in-depth analysis of why RCNN can be used as a “complement” to improve Faster RCNN. Based on our findings, we provide a simple yet effective decoupled classification refinement (DCR) module, that can be easily added to any current state-of-the-art object detectors to provide performance improvements.

### 4.1 Learning from RCNN Design

We train a modified RCNN with ResNet-50 as backbone and Faster RCNN predictions as region proposals. We find that with RCNN along, the detection result is deteriorated. Since RCNN does not modify box coordinate, the inferior result means worse classification. We find that many boxes having small intersections with an object are classified as that object instead of the background which Faster RCNN predicts. Based on this finding, we hypothesize that the drawback of RCNN is mainly root from that classification model is pre-trained without awaring object location. Since ResNet-50 is trained to be translation-invariance on ImageNet in multi-crop manner, no matter how much the intersection of the crop to the object is, classifier is encouraged to predict that class. This leads to the classifier in RCNN being “too strong” for proposal classification, and this is why RCNN needs a carefully tuned sampling strategy, *i.e.* a ratio of 1:3 of  $fg$  to  $bg$ . Straightforwardly, we are interested whether RCNN is “strong” enough to correct hard negatives. We make a minor modification to multiply RCNN classification score with Faster RCNN classification score and observe a boost of 1.9% (from 79.8% to 81.7%)! Thus, we consider that RCNN can be seen as a compliment of Faster RCNN in the following sense: the classifier of Faster RCNN is weaker but aware of object location whereas the classifier of RCNN is unaware of object location but stronger. Based on our findings, we propose the following three principals to design a better object detector.

#### 4.1.1 Decoupled Features

Current detectors still place classification network and localization network on the same backbone, hence we propose that classification head and localization head should

not share parameter (as the analysis given in Section 3.1), resulted in a decoupled feature using pattern by RCNN.

To demonstrate it is necessary to decouple classification and localization network, we explore sharing different amount of features between classification and localization. More specifically, we use two same networks, one for classification and one for localization and experiment with the number of stages that are shared (a “stage” is a group of features that have the same resolution). Results are shown in Table 2 (a). If all features are shared, the performance is 78.4% mAP. When we share less features, the performance monotonically increases to 83.0% when only the first stage is shared.

#### 4.1.2 Decoupled Optimization

RCNN also decouples the optimization for object proposal and classification. In this paper, we make a small change in optimization. We propose a novel two-stage training where, instead of optimizing the sum of classification and localization loss, we optimize the concatenation of classification and localization loss,  $L_{detection} = [L_{cls} + L_{bbox}, L_{cls}]$ , where each entry is being optimized independently in two steps.

#### 4.1.3 Adaptive Receptive Field

The most important advantage of RCNN is that its receptive field always covers the whole ROI, *i.e.* the receptive field size adjusts according to the size of the object by cropping and resizing each proposal to a fixed size. We agree that context information may be important for precise detection, however, we conjuncture that different amount of context introduced by fixed receptive filed might cause different performance to different sizes of objects. It leads to our last proposed principal that a detector should an adaptive receptive field that can change according to the size of objects it attends to. In this principal, the context introduced for each object should be proportional to its size, but how to decide the amount of context still remains an open question to be studied in the future. Another advantage of adaptive receptive field is that its features are well aligned to objects. Current detectors make predictions at high-level, coarse feature maps usually having a large stride, *e.g.* a stride of 16 or 32 is used in Faster RCNN, due

to sub-sampling operations. The sub-sampling introduces unaligned features, *e.g.* one cell shift on a feature map of stride 32 leads to 32 pixels shift on the image, and defects the predictions. With adaptive receptive field, the detector always attends to the entire object resulting in an aligned feature to make predictions. RCNN gives us a simple way to achieve adaptive receptive field, but how to find a more efficient way to do so remains an interesting problem needs studying.

To verify the importance of adaptive receptive field, we perform experiments by adding ROI Pooling to different stages in the parallel classification network. Results are shown in Table 2 (b). When we place the ROI Pooling at last stage, the network is less likely to adjust receptive field according to the ROI size and the performance is only 79.8%. However, as we place the ROI Pooling to earlier stages, the performance increases monotonically to 83.0%.

#### 4.2 DCR V1: A Naïve Method

Following these principals, we propose a naïve module (DCR V1) that can be easily augmented to Faster RCNN as well as any object detector to build a stronger detector. The overall pipeline is shown in Fig 4 (left). The green part and the orange part are the original Faster RCNN and our proposed DCR module, respectively. In particular, DCR V1 mainly consists a crop-resize layer and a strong classifier. The crop-resize layer takes two inputs, the original image and boxes produced by Faster RCNN, crops boxes on the original image and feeds them to the strong classifier after resizing them to a predefined size. Region scores of DCR V1 (Classifier 2) is aggregated with region scores of Faster RCNN (Classifier 1) by element-wise product to form the final score of each region. **In DCR V1, the two parts are trained separately and the scores are only combined during test time.**

The classification network in DCR V1 does not share any features with the detector backbone in order to preserve the quality of classification-aimed translation invariance feature. Furthermore, there is no error propagation between the classification network in DCR V1 and the base detector, thus the optimization of one loss does not affect the other. This in turn results in a decoupled pattern where the base detector is focused more on localization whereas the DCR V1 focuses more on classification. DCR V1 introduces adaptive receptive field by resizing boxes to a predefined size. Noticed that this processing is very similar to moving an ROI Pooling from final feature maps to the image, however, it is quite different than doing ROI Pooling on feature maps. Even though the final output feature map sizes are the same, features from ROI Pooling sees larger region because objects embedded in an image has richer context. We truncated the context by cropping objects directly on the image and the network cannot see context outside object regions.

#### 4.3 DCR V2: A Faster DCR Module

Although DCR V1 solves the problem of hard false positives, it also introduces extra computation overhead, including:

- 1) Cropping and resizing a large number of boxes on the original image.

- 2) Forward a large batch (usually 300 in Faster RCNN) of images to a deep network (a 152-layer ResNet).

The above computation overhead causes the real run time of DCR V1 module (1.3900 seconds/image) to be more than 100 times of that of the original Faster RCNN (0.0855 seconds/images). This number does not count the cropping and resizing time, which takes around 1~2 seconds per image with a sequential CPU implementation.

Inspired by [2], [20], we design a faster DCR module (DCR V2) that alleviates the computation overhead of DCR V1 module, shown in Fig 4 (right). That is, we solve (1) by using a highly paralleled GPU implementation of ROI Pooling and (2) by sharing part of the computation on the entire image. More specifically, we use a shared backbone network to extract high level features of the image and build the base detector and DCR V2 on top of the shared feature extractor. This design is based on the assumption that early stages of deep convolutional neural networks mainly extract low-level features (*e.g.* edges and textures) and we assume these low-level features can be shared among different tasks. The base detector is the same as that in the Faster RCNN and DCR V2 module is a **deep** convolutional classifier on regional features which are pooled by ROI Pooling.

To avoid the problem of feature sharing of classification and localization, we use a very deep residual network as the feature extractor of DCR V2. By placing this network on top of regional feature, the DCR V2 module is capable of learning translation invariance features for classifying regions. To introduce adaptive receptive field, we place the DCR V2 module at the early stage (by early stage, we mean layers that is close to input image). The early stages of network can learn texture-aware features that can be shared among different tasks as it has small stride and local receptive field. In our experiments, we also find it is beneficial to place DCR V2 module at early stage of the network.

Although sharing part of the regional feature extraction reduces the computation, we still need to process a large batch of ROIs during inference which requires a lot of memory and computation. To further reduce the inference time, we propose a simple yet efficient strategy called “top-sampling”. We find that the higher confidence score a false positive has, the larger impact it has (Fig. 1), thus we place different importance on false positives based on their confidence scores. In “top-sampling”, we only sample part of the detections whose confidence scores are within the top  $p$  percent which can further reduce the inference time while preserving the accuracy. For example, if we choose  $p = 50\%$ , then during the inference, we first score detections based on their maximum softmax scores and only pass top 50% boxes into the DCR V2 module.

The “top-sampling” strategy is also based on the fact that hard false positives that cannot be suppressed by post-processing (*e.g.* NMS) usually have high confidence, while false positives with low confidence are less likely to be suppressed. In this way, we can achieve a speed-accuracy trade-off by “attending” to detections with high confidence scores. Experimentally, we demonstrate that processing only the top 50% boxes degrades performance by less than 0.5% while nearly halving the actual run time.



## 4.4 Training

### 4.4.1 Training DCR V1

Since there is no error propagates from the DCR module to Faster RCNN, we train our object detector in a two-step manner. First, we train Faster RCNN to converge. Then, we train our DCR module on mini-batches sampled from hard false positives of Faster RCNN. Parameters of DCR module are pre-trained by ImageNet dataset [31]. We follow the image-centric method [2] to sample  $N$  images with a total mini-batch size of  $R$  boxes, *i.e.*  $R/N$  boxes per image. We use  $N = 1$  and  $R = 32$  throughout experiments. We use a different sampling heuristic that we sample not only foreground and background boxes but also hard false positive **uniformly**. Because we do not want to apply any prior knowledge to impose unnecessary bias on classifier. However, we observed that boxes from the same image have little variance. Thus, we fix Batch Normalization layer with ImageNet training set statistics. The newly added linear classifier (fully connected layer) is set with 10 times of the base learning rate since we want to preserve translation invariance features learned on the ImageNet dataset.

### 4.4.2 Training DCR V2

We train our DCR V2 module with the base detector in an end-to-end manner. The training of the base detector is the same as [3] and we mainly discuss the training of DCR V2 in detail. During training, we first sample  $R$  boxes from the outputs of the bounding box regression branch in the base detector, then we use these boxes as ROIs to extract regional features in the DCR V2 module. And the training of DCR V2 module is simply minimizing the cross entropy loss. In our experiments, we follow the optimal sampling strategy and label assignment that is used for training DCR V1. The final loss term is:

$$\mathcal{L} = \mathcal{L}_{\text{RPN}} + \mathcal{L}_{\text{RCNN}} + \mathcal{L}_{\text{DCRV2}}$$

## 4.5 Differences with Other Sampling Methods

### 4.5.1 RCNN

There are two major differences between our DCR module and RCNN. First, our DCR module is an end-to-end classifier. We use softmax classifier on top of the CNN feature where as RCNN trains another SVM using CNN features. Second, the motivation is different. The purpose of RCNN is to classify each region, but the purpose of DCR module is to correct false positives produced by base detectors. The difference in motivation results in different sampling heuristic. RCNN samples a large batch of foreground and background with some fixed ratio to achieve a good balance for training classifier. Our DCR module not only samples foreground and background, but also pay attention to samples that Faster RCNN makes “ridiculous” mistakes (hard false positives).

### 4.5.2 Hard Example Selection in Deep Learning

Hard example mining is originally used for optimizing SVMs to achieve the global optimum. In [32], an Online Hard Example Mining (OHEM) algorithm is proposed to

train Fast RCNN. Instead of sampling the minibatch randomly, [32] samples ROIs that have the top losses (sum of classification and localization loss) with respect to the current set of parameters. [22] uses a similar online approach, but instead of using hard examples with largest losses, [22] further imposes a restriction on the ratio of foreground and background in hard examples. The main difference is that hard examples may not always be hard false positives. DCR module focuses all its attention to deal with hard false positive which means it is more task-specific than hard example selection methods. Another difference between OHEM and our approach is that OHEM take both classification and localization loss into account whereas DCR module only considers classification.

### 4.5.3 Focal Loss (FL)

FL [33] is designed to down-weight the loss of well-classified examples by adding an exponential term related to the probability of ground truth class, *i.e.*  $\text{FL}(p_t) = -(1 - p_t)^\gamma \log(p_t)$ , where  $\gamma$  is a tunable parameter specifying how much to down-weight. The motivation of FL is to use a dense set of predefined anchors on all possible image locations without region proposals as well as the sampling step. Since background dominants in this large set of boxes, FL ends up down-weighting most of losses for backgrounds instead of focusing on hard false positives.

## 4.6 Difference with Other Cascade Methods

CRAFT [34] or Cascade RCNN [4] can be categorized as using cascade classifiers to improve object detection, but they are essentially different from our approach. In particular:

- 1) **The motivations are different.** CRAFT is motivated by “divide and conquer” philosophy, it is designed to split one tasks into several small tasks. Cascade RCNN is motivated by the fact that previous training methods (use IOU 0.5 to assign foregrounds) are misaligned with testing metrics (AP with IOU 0.5-0.95). Our motivation is that Faster RCNN is making many classification errors and we want to improve it by correctly classifying those misclassified regions.
- 2) **The architecture design principals are different.** Both CRAFT and Cascade RCNN have fixed receptive field and share the backbone features. DCR uses a novel “adaptive receptive field”, we name it “adaptive” because the effective receptive field is always equal to the size of the correspondent object proposal. The classifier in DCR uses different feature from object detector.
- 3) **The training and inference strategies are different.** All three method train the cascade classifier based on the output of the previous classifier. CRAFT uses *all* output except background objects of the previous classifier and uses *same* IOU threshold (*i.e.* 0.5) to assign labels. Cascade RCNN uses *all* output but reassign foreground labels based on *different* IOU thresholds (*i.e.* 0.5, 0.6, 0.7). DCR uses only *part* of the previous output with a novel sampling heuristic (hard false positive sampling). In inference, both CRAFT and Cascade RCNN use prediction of the

last classifier, their hypothesis is the cascade classifier is always better than the previous one. However, DCR uses *both* classifiers, our hypothesis is that our classifiers complement each other.

## 5 EXPERIMENTS

### 5.1 Implementation Details

#### 5.1.1 DCR V1

We train base detectors, *e.g.* Faster RCNN, following their original implementations. We use default settings in 4.4 for DCR module, we use ROI size  $224 \times 224$  and use a threshold of 0.3 to identify hard false positives. Our DCR module is first pre-trained on the ILSVRC 2012 dataset [31]. In fine-tuning, we set the initial learning rate to 0.0001 *w.r.t.* one GPU and weight decay of 0.0001. We follow linear scaling rule in [35] for data parallelism on multiple GPUs and use 4 GPUs for PASCAL VOC and 8 GPUs for COCO. Synchronized SGD with momentum 0.9 is used as optimizer. No data augmentation except horizontal flip is used.

#### 5.1.2 DCR V2

The implementations of Faster RCNN is the same as their original ones and we will only discuss detail implementation of our DCR V2 module using a 101-layer ResNet as example.

Fig 5 (a) shows a detailed block diagram of default DCR V2 module. Follow the naming convention in mainstream frameworks' implementation of ResNet, *e.g.* MXNET and PyTorch, we denote the five stages of ResNet as conv1 (the first Convolution, BatchNorm and Max Pooling), Stage1 (the first residual stage containing 3 residual blocks), Stage2 (the second residual stage containing 4 residual blocks), Stage3 (the third residual stage containing 23 residual blocks) and Stage4 (the fourth residual stage containing 3 residual blocks). Following [13], we append an additional  $3 \times 3$  convolution with 256 output channel after Stage4. We place the RPN at the end of Stage3 and the ROI Pooling that takes proposals from RPN as input at the end of Stage4. For the RCNN-head, we follow [13] to use two MLP with 1024 hidden layers followed by a classification branch and a bounding box regression branch. To construct the DCR V2 module, we place another ROI Pooling at the end of Stage1 that takes detections of RCNN-head as ROI input. On top of the ROI Pooling, we simply copy Stage2, Stage3, Stage4 of ResNet and add a global average pooling followed with a linear classifier. We initialize both Faster RCNN and DCR V2 module with same ImageNet pretrained weights.

We use an initial learning rate of 0.0005 and a batchsize of 1 for each GPU. The weight decay is set to 0.0005 and momentum is set to 0.9. We only use horizontal flip during training.

### 5.2 Ablation Studies on PASCAL VOC

We comprehensively evaluate our method on the PASCAL VOC detection benchmark [15]. We use the union of VOC 2007 trainval and VOC 2012 trainval as well as their horizontal flip as training data and evaluate results on the VOC 2007 test set. We primarily evaluate the detection mAP with IoU 0.5 (mAP@0.5). Unless otherwise stated, all ablation studies

are performed with ResNet-101 for Faster RCNN, ResNet-50 as classifier for our DCR V1 module and ResNet-101 for DCR V2 module.

#### 5.2.1 Ablation study on sampling heuristic

We compare results with different sampling heuristic in training DCR module:

- random sample: a minibatch of ROIs are randomly sampled for each image
- hard false positive only: a minibatch of ROIs that are hard positives are sampled for each image
- hard false positive and background: a minibatch of ROIs that are either hard positives or background are sampled for each image
- hard false positive and foreground: a minibatch of ROIs that are either hard positives or foreground are sampled for each image
- hard false positive, background and foreground: the difference with random sample heuristic is that we ignore easy false positives during training.
- RCNN-like: we follow the Fast RCNN's sampling heuristic, we sample two images per GPU and 64 ROIs per image with  $fg:bg=1:3$ .

Results are shown in Table 1 (a). We find that the result is insensitive to sampling heuristic. Even with random sampling, an improvement of 2.0% in mAP is achieved. With only hard false positive, the DCR achieves an improvement of 1.6% already. Adding foreground examples only further gains a 0.2% increase. Adding background examples to false negatives harms the performance by a large margin of 1.1%. We hypothesize that this is because comparing to false positives, background examples dominating in most images results in a classifier bias to predicting background. This finding demonstrate the importance of hard negative in DCR training. Unlike RCNN-like detectors, we do not make any assumption of the distribution of hard false positives, foregrounds and backgrounds. To balance the training of classifier, we simply uniformly sample from the union set of hard false positives, foregrounds and backgrounds. This uniform sample heuristic gives the largest gain of 2.5% mAP. We also compare our training with RCNN-like training. Training with RCNN-like sampling heuristic with  $fg:bg=1:3$  only gains a margin of 1.9%.

#### 5.2.2 Ablation study on other hyperparameters

We compare results with different threshold for defining hard false positive: [0.2, 0.25, 0.3, 0.35, 0.4]. Results are shown in Table 1 (b). We find that the results are quite insensitive to threshold of hard false positives and we argue that this is due to our robust uniform sampling heuristic. With hard false positive threshold of 0.3, the performance is the best with a gain of 2.5%.

We also compare the influence of size of sampled ROIs during training: [8, 16, 32, 64]. Results are shown in Table 1 (c). Surprisingly, the difference of best and worst performance is only 0.3%, meaning our method is highly insensitive to the sampling size. With smaller sample size, the training is more efficient without severe drop in performance.



Sample method	mAP	FP Score	mAP	Sample size	mAP	ROI scale	mAP	Test Time
Baseline	79.8	Baseline	79.8	Baseline	79.8	Baseline	79.8	0.0855
Random	81.8	0.20	82.2	8 Boxes	82.0	$56 \times 56$	80.6	0.0525
FP Only	81.4	0.25	81.9	16 Boxes	82.1	$112 \times 112$	82.0	0.1454
FP+FG	81.6	0.30	<b>82.3</b>	32 Boxes	<b>82.3</b>	$224 \times 224$	<b>82.3</b>	0.5481
FP+BG	80.3	0.35	82.2	64 Boxes	82.1	$320 \times 320$	82.0	1.0465
FP+FG+BG	<b>82.3</b>	0.40	82.0					
RCNN-like	81.7							

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

DCR Depth	mAP	Test Time	Base detector	mAP	Model capacity	mAP
Baseline	79.8	0.0855	Faster	79.8	Faster w/ Res101	79.8
18	81.4	0.1941	Faster+DCR	82.3	Faster w/ Res152	80.3
34	81.9	0.3144	DCN	81.4	Faster Ensemble	81.1
50	82.3	0.5481	DCN+DCR	83.2	Faster w/ Res101+DCR-50	82.3
101	82.3	0.9570				
152	<b>82.5</b>	1.3900				

(e) (f) (g)

TABLE 1: Ablation studies results. Evaluate on PASCAL VOC2007 test set. Baseline is Faster RCNN with ResNet-101 as backbone. DCR module uses ResNet-50. (a) Ablation study on sampling heuristics. (b) Ablation study on threshold for defining hard false positives. (c) Ablation study on sampling size. (d) Ablation study on ROI scale and test time (measured in seconds/image). (e) Ablation study on depth of DCR module and test time (measured in seconds/image). (f) DCR module with difference base detectors. Faster denotes Faster RCNN and DCN denotes Deformable Faster RCNN, both use ResNet-101 as backbone. (g) Comparison of Faster RCNN with same size as Faster RCNN + DCR.

DCR V2 Stage	mAP	Test Time	ROI Pooling Stage	mAP	Test Time	top-p	mAP	Test Time
Baseline	79.8	0.0855	Baseline	79.8	0.0855	DCR V1	82.5	1.3900
Stage 4	78.4	0.0945	Stage 3 (after DCR V2)	79.8	0.1474	100%	83.0	0.7929
Stage 3	80.0	0.1543	Stage 2	80.7	0.2027	75%	82.9	0.6323
Stage 2	82.8	0.6540	Stage 1	82.9	0.6590	50%	82.8	0.4653
Stage 1	<b>83.0</b>	0.7929	Stage 0 (before DCR V2)	<b>83.0</b>	0.7929	25%	81.9	0.3015
						0%	80.2	0.0855

(a) (b) (c)

TABLE 2: Ablation studies results of DCR V2 Module. Evaluate on PASCAL VOC2007 test set. Baseline is Faster RCNN with ResNet-101 as backbone. (a) Ablation study on the amount of feature sharing by adding DCR V2 branch after different stages in ResNet-101 backbone, ROI Pooling is placed before DCR V2 module. (b) Ablation study on adaptive receptive field by placing ROI Pooling after different stages of DCR V2 module, DCR V2 module is placed after Stage 1 of ResNet-101. (c) DCR V2 Top-Sampling during inference. Baseline model is Faster RCNN with ResNet-101 backbone. For the DCR V2 module, we place it at the end of Stage 1. The speed is measured on a single NVIDIA GTX 1080 TI.

### 5.2.3 Ablation study for DCR V2

We perform two ablation studies for our DCR V2 module: 1. after which stage to add DCR V2 module in the backbone? 2. where to add ROI Pooling to extract regional feature in the DCR V2 module? Results are shown in Table 2 (a) and (b).

We experiment on adding DCR V2 **after** Stage1, Stage2, Stage3, Stage4 in ResNet-101 to study the effect of feature sharing between Faster RCNN and DCR V2, and we simply use the remaining stage(s) that are not shared for DCR V2 module (Fig 5 (b)). In this case, ROI Pooling for DCR V2 is placed before DCR V2 module, *i.e.* DCR V2 module acts directly on the regional feature. As we place DCR V2 to earlier stages, we let Faster RCNN share less features with DCR V2 module and we observe an increase in the performance (mAP is increased from 78.4 to 83.0 as we place DCR V2 from Stage4 to Stage1). However, the inference time increases as expected. These results are consistent with our hypothesis in Section 3.1 that feature sharing between classification and localization might be harmful.

When the position of DCR V2 module is fixed (after Stage1 of the ResNet), we also explore the effect of ROI Pooling position within DCR V2 (Fig 5 (c)). The position of ROI Pooling decides a trade-off between computing regional feature and image-level feature. If we place ROI

Pooling at Stage3 (the last stage) of DCR V2 (after the module), all features are computed at image level and the mAP is only 79.8. If we place ROI Pooling at Stage0 (before DCR V2 module), then all features of DCR V2 are computed at ROI level, the mAP is 83.0. This results are consistent with our hypothesis in Section 3.3 that the model should have adaptive receptive field, which means, placing ROI Pooling at earlier stage is beneficial.

### 5.2.4 Speed and accuracy trade-off

There are in general two ways to reduce inference speed, one is to reduce the size of input and the other one is to reduce the depth of the network. We compare 4 input sizes:  $56 \times 56$ ,  $112 \times 112$ ,  $224 \times 224$ ,  $320 \times 320$  as well as 5 depth choices: 18, 34, 50, 101, 152 and their speed. Results are shown in Table 1 (d) and (e). The test speed is linearly related to the area of input image size and there is a severe drop in accuracy if the image size is too small, *e.g.*  $56 \times 56$ . For the depth of classifier, deeper model results in more accurate predictions but also more test time. We also notice that the accuracy is correlated with the classification accuracy of classification model, which can be used as a guideline for selecting DCR module.

Results of our DCR V2 with different running times are shown in Table 2 (c). We evaluate the running time with

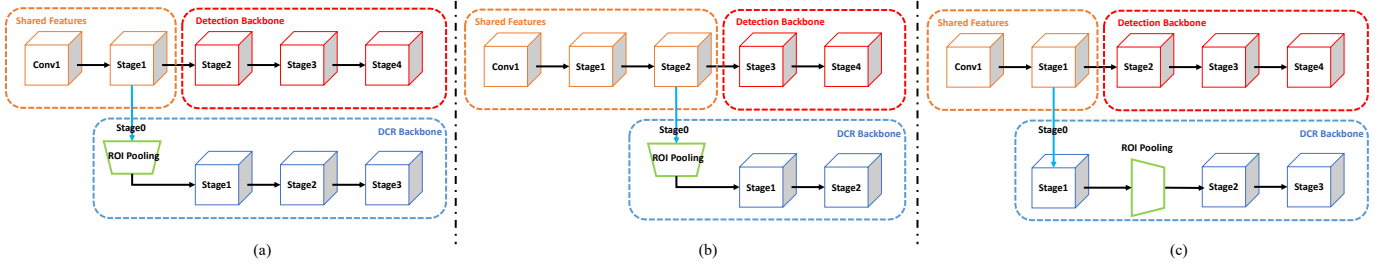


Fig. 5: Illustration of ablation studies performed on DCR V2. For simplicity, detection head including RPN, ROI Pooling, cls and bbox regression are ignored. (a) Our default DCR V2 structure. (b) Ablation study on the amount of features to share. This is an example when DCR V2 is connected after Stage2 of ResNet with mAP 82.8%. (c) Ablation study on adaptive receptive field of DCR V2. This is an example when ROI Pooling is placed after Stage1 of DCR V2 with mAP 82.9%.

DCR V2 after Stage1 on a single NVIDIA GTX 1080 TI GPU. Comparing with DCR V1 which has a run time of 1.3900 seconds/image, DCR V2 only requires 0.7929 seconds/image to get even better performance (+0.5% mAP) which is 1.75 times faster than DCR V1. If we use the “top-sampling” strategy to only sample top 50% of the detections, the run time becomes 0.4653 seconds/image with a degradation of 0.2% in mAP, which is 3 times faster than DCR V1 and 1.7 times faster than DCR V2 with out sampling. Moreover, as mentioned earlier, the actual run time is much longer than 1.3900 seconds/image as we do not count the running time to crop and resize 300 images on the original image. That is, DCR V2 has a much better speed/accuracy trade-off over DCR V1.

### 5.2.5 Generalization to more advanced object detectors

We evaluate the DCR V1 module on Faster RCNN and advanced Deformable Convolution Nets (DCN) [14]. Results are shown in Table 1 (f). Although DCN is already among one of the most accurate detectors, its classifier still produces hard false positives and our proposed DCR module is effective in eliminating those hard false positives. More results of DCR V2 on advanced detectors are shown in Table 3, 4, 5 and 6.

### 5.2.6 Where is the gain coming from?

One interesting question is where the accuracy gain comes from. Since we add a large convolutional network on top of the object detector, does the gain simply comes from more parameters? Or, is DCR an ensemble of two detectors? To answer this question, we compare the results of Faster RCNN with ResNet-152 as backbone (denoted Faster-152) and Faster RCNN with ResNet-101 backbone + DCR-50 (denoted Faster-101+DCR-50) and results are shown in Table 1 (g). Since the DCR module is simply a classifier, the two network have approximately the same number of parameters. However, we only observe a marginal gain of 0.5% with Faster-152 while our Faster-101+DCR-50 has a much larger gain of 2.5%. To show DCR is not simply then ensemble to two Faster RCNNs, we further ensemble Faster RCNN with ResNet-101 and ResNet-152 and the result is 81.1% which is still 1.1% worse than our Faster-101+DCR-50 model. This means that the capacity does not merely come from more parameters or ensemble of two detectors.

## 5.3 PASCAL VOC Results

### 5.3.1 VOC 2007

We use a union of VOC2007 trainval and VOC2012 trainval as training set and test the model on VOC2007 test set. We use the default training setting and ResNet-152 as classifier for the DCR V1 and ResNet-101 for DCR V2 module. We train our model for 7 epochs and reduce learning rate by  $\frac{1}{10}$  after 4.83 epochs. Results are shown in Table 3. Notice that based on DCN as base detector, our single DCR module achieves competitive result of 84.2% without using extra data (e.g. COCO data), multi scale training/testing, ensemble or other post processing tricks.

### 5.3.2 VOC 2012

We use a union of VOC2007 trainvaltest and VOC2012 trainval as training set and test the model on VOC2012 test set. We use the same training setting of VOC2007. Results are shown in Table 4. Based on DCN and DCR module, our model is the first model achieves over 81.0% on the VOC2012 test set. A competitive result of 81.2% is achieved using only single model, without any post processing tricks.

## 5.4 COCO Results

All experiments on COCO follow the default settings and use ResNet-152 for DCR V1 and ResNet-101 for DCR V2 module. We train our model for 8 epochs on the COCO dataset and reduce the learning rate by  $\frac{1}{10}$  after 5.33 epochs. We report results on two different partition of COCO dataset. One partition is training on the union set of COCO2014 train and COCO2014 val35k together with 115k images (this is the same as COCO2017 train) and evaluate results on the COCO2014 minival with 5k images held out from the COCO2014 val (this is the same as COCO2017 val). The other partition is training on the standard COCO2014 trainval with 120k images (this is the same as COCO2017 trainval) and evaluate on the COCO test-dev by submitting results to the COCO evaluation server. We use Faster RCNN [3], Feature Pyramid Networks (FPN) [13] and the Deformable ConvNets [14] as base detectors.

### 5.4.1 COCO2014 minival (COCO2017 val)

Results are shown in Table 5. DCR V2 consistently outperforms DCR V1. Our DCR V2 module improves Faster RCNN by 3.6% from 30.0% to 33.6% in COCO AP metric. Faster RCNN with DCN is improved by 3.1% from 34.4% to





four categories: (1) Loc: IOU with ground truth boxes is in the range of  $[0.1, 0.5]$ ; (2) Sim: detections have at least 0.1 IOU with objects in predefined similar classes, *e.g.* dog and cat are similar classes; (3) Oth: detections have at least 0.1 IOU with objects not in predefined similar classes; (4) BG: all other false positives are considered background. We observe that comparing with Faster RCNN, DCR module has much larger ratio of localization error and the number of false positives is greatly reduced on some classes, *e.g.* in the animal class, the number of false positives is largely reduced by 4 times and initial percentage of localization error increases from less than 30% to over 50%. This statistics are consistent with motivations to reducing classification errors by reducing number of false positives.

Fig 6 (b) compares the sensitivity of Faster RCNN and DCR to object characteristics. [42] defines object with six characteristics: (1) occ: occlusion, where an object is occluded by another surface; (2) trn: truncation, where there is only part of an object; (3) size: the size of an object measure by the pixel area; (4) asp: aspect ratio of an object; (5) view: whether each side of an object is visible; (6) part: whether each part of an object is visible. Normalized AP is used to measure detectors performance and more details can be found in [42]. In general, the higher the normalized AP, the better the performance. The difference between max and min value measure the sensibility of a detector, the smaller the difference, the less sensible of a detector. We observe that DCR improves normalized AP and sensitivity on all types of object and improves sensitivity significantly on occlusion and size. This increase came from the adaptive field of DCR, since DCR can focus only on the object area, making it less sensible to occlusion and size of objects.

## 6 VISUALIZATION

We visualize all false positives with confidence larger than 0.3 for both Faster RCNN and our DCR module in Fig 7. We observe that the DCR module successfully suppresses all three kinds of hard false positives to some extends.

The first image shows reducing the first type of false positives (part of objects). Faster RCNN (top) classifies the head of the cat with a extremely high confidence (0.98) but it is eliminated by the DCR module.

The second to the fourth images demonstrate situations of second type of false positives (similar objects) where most of false positives are suppressed ("car" in the second image and "horse" in the third image). However, we find there still exists some limitations, *e.g.* the "dog" in the third image where it is supposed to be a cow and the "person" in the fourth image. Although they are not suppressed, their scores are reduced significantly ( $0.96 \rightarrow 0.38$  and  $0.92 \rightarrow 0.52$  respectively) which will also improve the overall performance. It still remains an open questions to solve these problems. We hypothesize that by using more training data of such hard false positives (*e.g.* use data augmentation to generate such samples).

The last image shows example for the third type of false positives (backgrounds). A part of background near the ground truth is classified as a "boat" by the Faster RCNN and it is successfully eliminated by our DCR module.

## 7 CONCLUSION

In this paper, we analyze error modes of state-of-the-art region-based object detectors and study their potentials in accuracy improvement. We hypothesize that good object detectors should be designed following three principles: decoupled features, decoupled optimization and adaptive receptive field. Based on these principles, we propose a simple, effective and widely-applicable DCR module that achieves new state-of-the-art. Furthermore, we use experiments to support that a good detector should have decoupled features and adaptive receptive field and we observe an interesting novel trade-off between feature sharing and speed/accuracy. In the future, we will further study what architecture makes a good object detector, adaptive feature representation in multi-task learning, and efficiency improvement of our DCR module. We hope this paper could motivate the community to study new directions of improving current object detection frameworks. Specifically, (1) DCR module still has 4.3% mAP to be improved due to false positives and how to suppress these remaining false positives remains an open question. (2) how to design more efficient decoupled structure is yet another interesting research direction.

## ACKNOWLEDGMENTS

This work is in part supported by IBM-ILLINOIS Center for Cognitive Computing Systems Research (C3SR) - a research collaboration as part of the IBM AI Horizons Network; and by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/ Interior Business Center (DOI/IBC) contract number D17PC00341. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DOI/IBC, or the U.S. Government. We thank Jiashi Feng for helpful discussions.

## REFERENCES

- [1] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE CVPR*, 2014, pp. 580–587.
- [2] R. Girshick, "Fast r-cnn," in *IEEE ICCV*, 2015, pp. 1440–1448.
- [3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *NIPS*, 2015, pp. 91–99.
- [4] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," in *IEEE CVPR*, June 2018.
- [5] H. Xu, X. Lv, X. Wang, Z. Ren, and R. Chellappa, "Deep regionlets for object detection," *ECCV*, 2017.
- [6] J. Li, Y. Wei, X. Liang, J. Dong, T. Xu, J. Feng, and S. Yan, "Attentive contexts for object detection," *IEEE Transactions on Multimedia*, vol. 19, no. 5, pp. 944–954, 2017.
- [7] J. Li, X. Liang, J. Li, Y. Wei, T. Xu, J. Feng, and S. Yan, "Multistage object detection with group recursive learning," *IEEE Transactions on Multimedia*, vol. 20, no. 7, pp. 1645–1655, 2018.
- [8] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, "Perceptual generative adversarial networks for small object detection," in *IEEE CVPR*, 2017.
- [9] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "Unitbox: An advanced object detection network," in *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 2016, pp. 516–520.

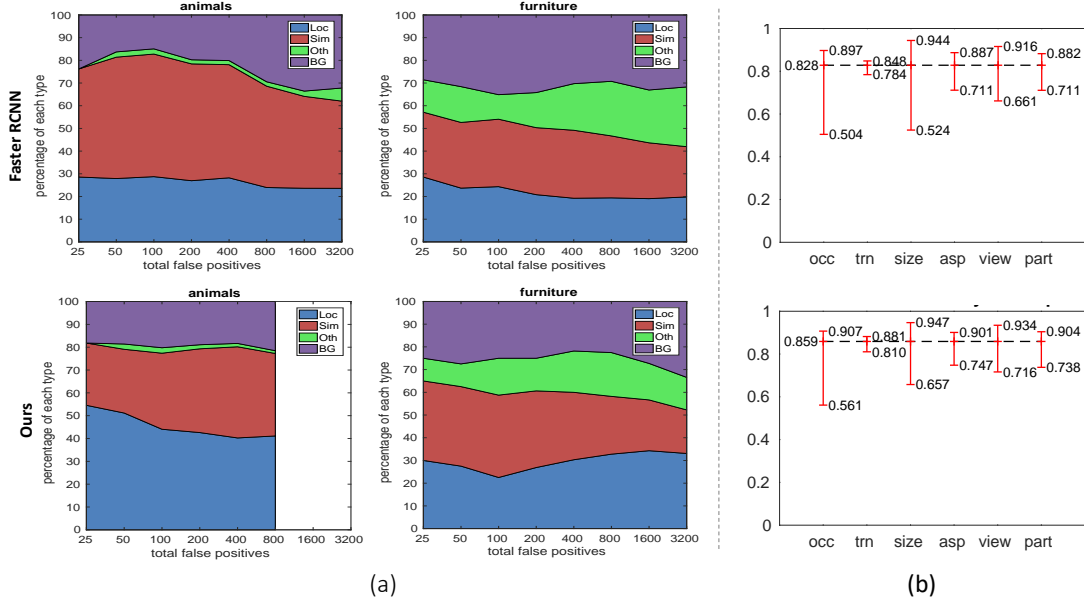


Fig. 6: Analysis results between Faster RCNN (top row) and our methods (bottom row) by [42]. Left of the dashed line: distribution of top false positive types. Right of the dashed line: sensitivity to object characteristics.

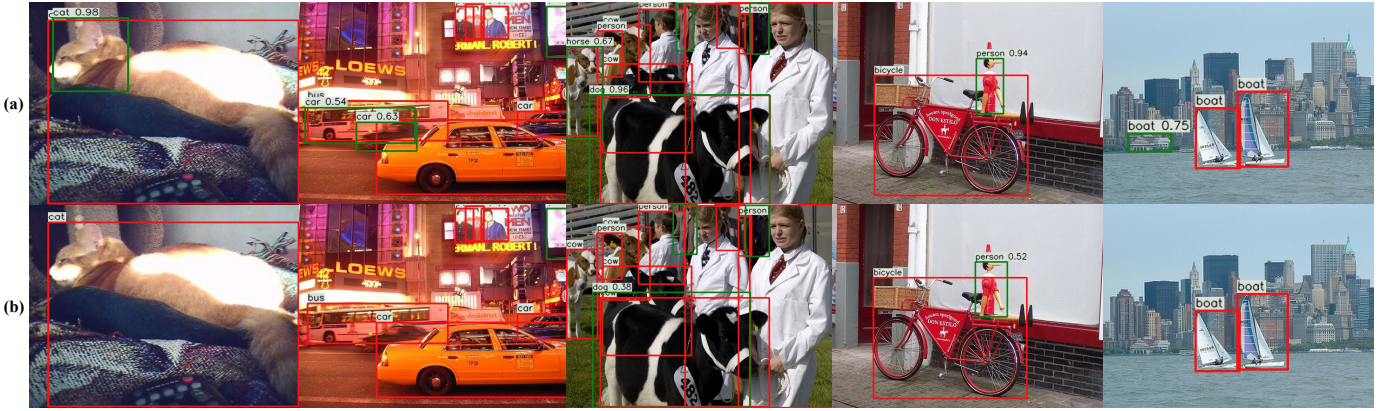


Fig. 7: Comparison of hard false positives with confidence score higher than 0.3 between Faster RCNN and Our methods. Red box: ground truth object. Green box: hard false positive. Row (a), results from Faster RCNN. Row (b), results of our DCR module.

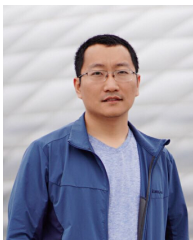
- [10] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei, "Relation networks for object detection," in *IEEE CVPR*, 2018.
- [11] Y. Wei, Z. Shen, B. Cheng, H. Shi, J. Xiong, J. Feng, and T. Huang, "Ts2c: Tight box mining with surrounding segmentation context for weakly supervised object detection," in *European Conference on Computer Vision*, 2018.
- [12] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *NIPS*, 2016, pp. 379–387.
- [13] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *IEEE CVPR*, vol. 1, no. 2, 2017, p. 4.
- [14] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *IEEE ICCV*, 2017, pp. 764–773.
- [15] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *IJCV*, vol. 88, no. 2, pp. 303–338, Jun. 2010.
- [16] 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*, 2014, pp. 740–755.
- [17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE CVPR*, 2016, pp. 770–778.
- [18] B. Cheng, Y. Wei, H. Shi, R. Feris, J. Xiong, and T. Huang, "Revisiting rcnn: On awakening the classification power of faster rcnn," in *ECCV*, 2018.
- [19] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *IJCV*, vol. 104, no. 2, pp. 154–171, 2013.
- [20] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *ECCV*, 2014, pp. 346–361.
- [21] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.
- [22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *ECCV*, 2016, pp. 21–37.
- [23] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "Dssd: Deconvolutional single shot detector," *arXiv preprint arXiv:1701.06659*, 2017.
- [24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE CVPR*, 2016, pp. 779–788.
- [25] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *IEEE CVPR*, 2017, pp. 6517–6525.
- [26] P. Viola and M. J. Jones, "Robust real-time face detection," *IJCV*, vol. 57, no. 2, pp. 137–154, 2004.



- [27] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.
- [28] L. Bourdev and J. Brandt, "Robust object detection via soft cascade," in *IEEE CVPR*, vol. 2, 2005, pp. 236–243.
- [29] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE TPAMI*, vol. 32, no. 9, pp. 1627–1645, 2010.
- [30] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," in *IEEE CVPR*, 2015, pp. 5325–5334.
- [31] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE CVPR*, 2009, pp. 248–255.
- [32] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *IEEE CVPR*, 2016, pp. 761–769.
- [33] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *IEEE ICCV*, 2017, pp. 2980–2988.
- [34] B. Yang, J. Yan, Z. Lei, and S. Z. Li, "Craft objects from images," in *IEEE CVPR*, 2016, pp. 6043–6051.
- [35] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, large minibatch sgd: training imagenet in 1 hour," *arXiv preprint arXiv:1706.02677*, 2017.
- [36] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *IEEE ICCV*, 2017, pp. 2980–2988.
- [37] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *IEEE CVPR*, 2017, pp. 5987–5995.
- [38] B. Singh and L. S. Davis, "An analysis of scale invariance in object detection-snip," in *IEEE CVPR*, 2018.
- [39] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun, "Detnet: Design backbone for object detection," in *ECCV*, September 2018.
- [40] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *ECCV*, September 2018.
- [41] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang, "Acquisition of localization confidence for accurate object detection," in *ECCV*, September 2018.
- [42] D. Hoiem, Y. Chodpathumwan, and Q. Dai, "Diagnosing error in object detectors," in *ECCV*, 2012, pp. 340–353.

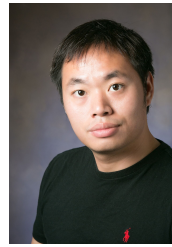


**Bowen Cheng** is a Ph.D. student in Electrical and Computer Engineering at University of Illinois Urbana-Champaign (UIUC). His Ph.D. advisor is Prof. Thomas Huang and he is doing research in computer vision. Specifically, he works on object detection, image classification and semantic segmentation. Before commencing his graduate studies, he received his B.S. in Electrical and Computer Engineering at UIUC in 2017.



**Yunchao Wei** is currently a Postdoctoral Researcher at the University of Illinois at Urbana-Champaign. He received his Ph.D. degree from Beijing Jiaotong University in 2016. He received Excellent Doctoral Dissertation Awards of Chinese Institute of Electronics (CIE) in 2016, the Winner prize of the object detection task (1a) in ILSVRC 2014, the Runner-up prizes of all the video object detection tasks in ILSVRC 2017. He has published more than 30 papers in top-tier conferences/journals, with over 1000 citations in

Google Scholar. His current research interest focuses on computer vision techniques for large-scale data analysis. Specifically, he has done work in weakly- and semi-supervised object recognition, multi-label image classification, image/video object detection, multi-modal analysis.



**Honghui Shi** is a Research Staff Member at IBM T. J. Watson Research Center, and a Graduate Faculty Member at Electrical and Computer Engineering (ECE) at the University of Illinois at Urbana-Champaign (UIUC). He received his Bachelor degree from Tsinghua University in 2009. After 3 years of experience in industry and as co-founder of an Internet startup, he returned to his pursuit in academia. He received MS in 2016 and PhD in 2017 both from ECE, UIUC. He had been working on and leading more than

10 academic research projects during his student time in various topics spanning visual recognition, financial modeling, medical image analysis and etc. He is PI/Co-PI of multiple industrial and academic projects with a focus on connecting cutting edge AI research with real world data and problems at scale. He is a member of the IEEE.



**Rogerio Schmidt Feris** is the head of computer vision and multimedia research at IBM T.J. Watson Research Center, as part of IBM Research AI. He joined IBM in 2006 after receiving a PhD in computer science from the University of California, Santa Barbara. He has also worked as an Affiliate Associate Professor at University of Washington and as an Adjunct Associate Professor at Columbia University, teaching courses on Visual Recognition and Search and Automatic Video Surveillance. He has authored over 100

technical papers and has over 40 issued patents in the areas of computer vision, multimedia, and machine learning. His work has been covered by ABC News and NY Times, among other media outlets. He has served as Area Chair of top premiere computer vision and machine learning conferences, such as CVPR, ICCV, and NIPS. He also serves as an Associate Editor of TPAMI. See more at <http://rogerioferis.com>.



**Jinjun Xiong** is a program director for cognitive computing systems research at the IBM T.J. Watson Research Center and a co-director of the IBM-Illinois Center for Cognitive Computing Systems Research. His research interests include cognitive computing, big data analytics, deep learning, smarter energy, and application of cognitive computing for industrial solutions. Xiong received a PhD in electrical engineering from the University of California, Los Angeles. He has received three Best Papers, one Best

Paper in Track, and eight Best Paper nominations. He is an IBM Master Inventor and has received various IBM technical achievement awards and IEEE Region One Outstanding Technical Contribution Award.



**Thomas Huang** received Sc.D. from the Massachusetts Institute of Technology in 1963. He was a tenured faculty member at MIT from 1963 to 1973. He was director of the Information and Signal Processing Laboratory at Purdue University from 1973 to 1980. He is currently Research Professor and Swanlund Endowed Chair Professor Emeritus with the University of Illinois at Urbana-Champaign, and Director of Image Formation and Processing (IFP) group at Beckman Institute. He has co-authored over 20 books and

over 1300 papers. He is a leading researcher in computer vision, image processing and multi-modal signal processing. He is a member of the National Academy of Engineering. He is a fellow of IEEE, IAPR, and OSA. He was a recipient of the IS&T and SPIE Imaging Scientist of the Year Award in 2006. He has received numerous awards, including the IEEE Jack Kilby Signal Processing Medal, the King-Sun Fu Prize of the IAPR, and the Azriel Rosenfeld Life Time Achievement Award at ICCV.