Decoupled R-CNN: Sensitivity-Specific Detector for Higher Accurate Localization

Dong Wang, Kun Shang, Huaming Wu, *Senior Member, IEEE,* Ce Wang, *Member, IEEE,*

Abstract—**Object detection, as a fundamental problem in computer vision, has been widely used in many industrial applications, such as intelligent manufacturing and intelligent video surveillance. In this work, we find that classification and regression have different sensitivities to the object translation, from the investigation about the availability of highly overlapping proposals. More specifically, the regressor head has intrinsic characteristics of higher sensitivity to translation than the classifier. Based on it, we propose a decoupled sampling strategy for a deep detector, named Decoupled R-CNN, to decouple the proposals sampling for the two tasks, which induces two sensitivity-specific heads. Furthermore, we adopt the cascaded structure for the single regressor head of Decoupled R-CNN, which is an extremely simple but highly effective way of improving the performance of object detection. Extensive empirical analyses using real-world datasets demonstrate the value of the proposed method when compared with the state-of-the-art models. The reproducing code is available at https://github.com/shouwangzhe134/Decoupled-R-CNN.**

Index Terms—**Object Detection, R-CNN, Two-stage Detection, Decoupled Sampling Strategy, Decoupled R-CNN.**

I. Introduction

Recent years have witnessed the computer vision systems (CVS) bring rapid development in the 'industry 4.0' era. Object detection is one of the fundamental problems in CVS, considered as an enabling technology contributing to the growth of many industrial applications. Furthermore, it provides advanced information for various downstream tasks. For example, the CVS in smart manufacturing detects the special tiny object to obtain better quality products that are available at lower costs. Person search [1] in an intelligent surveillance system provides the human-concerned semantic cue for adding to already-heightened security awareness. Hence, object detection has resulted in increased research interest in the application of data/predictive analytics.

Due to the remarkable performance of deep learning [2], especially convolutional neural network (CNN) [3], [4], the CSV has made breakthroughs in accuracy and speed to detect the concerned object. Generally, existing object detection methods can be categorized as: one-stage detection, such as YOLO series [5]–[7] and SSD series [8], [9], two-stage detection, such as R-CNN series [10]–[15], and even multi-stage detection, such as Cascade R-CNN [16]. Despite these different detection frameworks, object detection always aims to determine where the object locates and which category the object belongs to in a given image, which is constructed as a multi-task learning problem: regression and classification.

Specifically, the one-stage frameworks [17]–[20] directly use the CNN-based features to classify the objects in the predefined reference boxes and regress the offsets of refinements without a proposal generation step. However, the one-stage detector comes to terms with the bounding box regression, because it seems to be translation invariant[1]. For two-stage detectors [21], [22], which treat the detection as a coarse-to-fine process, they first generate region proposals (a pre-processing step) and then adopt a region of interest (RoI) pooling to extract a fixed-length representation for the next steps. It makes the extracted feature reveal the position information of the region proposal explicitly. So the region-specific RoI pooling breaks down translation-invariance to some extent. This phenomenon makes the two-stage detector to be more sensitive to translation than the one-stage.

Note that the mentioned two-stage detectors perform the classification and regression tasks on the same dense proposals generated by Region Proposal Networks (RPNs), where they have opposite preferences towards translation. Intuitively, the shift on an object inside an image should be indiscriminative for classification, while the translation of an object inside a candidate box should produce meaningful responses for regression. It led to a big dilemma: **increasing translation invariance for classification vs. respecting translation variance for regression.** More specifically, the translation-invariant of an object is beneficial to classification, and in contrast, the translation-variant is helpful for regression.

Unfortunately, the detector must distinguish the foreground objects from the background and assign accurate bounding boxes to different objects simultaneously, as shown in Fig. 1-(a). It is particularly difficult to accomplish the two tasks perfectly for the RoI-based detector with the shared proposals and heads for classification and regression. Note that the detectors based on deep learning are very example intensive. Training on the same proposals makes both the classifier and the regressor finally have correlated sensitivities to translation, especially for the shared heads. This process is particularly problematic for deep detectors: **two different tasks with the same sensitivity**. Based on the above, classification and

Dong Wang and Huaming Wu are with the Center of Applied Mathematics, Tianjin University, Tianjin 300072, China.

Kun Shang (corresponding author) is with Research Center for Medical AI, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China. Email: kunzzz.shang@gmail.com.

Ce Wang is with Institute of Computing Technology, CAS, Beijing, China and Suzhou Institute of Intelligent Computing Technology, CAS, Suzhou, China.

[1]There are two reasons: the misalignment between the reference box and the extracted feature (a 3x3 convolution centered at the reference box), and the shared convolutional feature among multiple reference boxes with the same center.

regression should be considered on specific RPN proposals, which is shown in Fig. 1-(b).

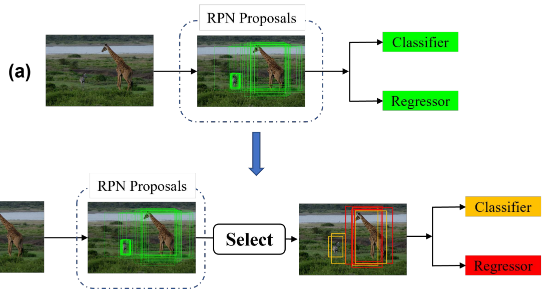

Fig. 1: The illustration of our motivation. (a) The regressor and the classifier share the same training samples. (b) The regressor and the classifier should be operated on the different views for improving the performance of the two-stage detector.

In this work, we meticulously revisit the sampling process in the RoI-based object detector to seek the solution of the sensitivity mismatching. Then, we introduce a novel network structure with the sensitivity-specific training scheme, named Decoupled R-CNN, to alleviate the aforementioned problems. Specifically, we propose a decoupled sampling strategy, that classification adopts the random sampling with a positive-to-negative ratio and regression adopts only positive sampling, to select the specific proposals with the different sensitivities to translation for the corresponding heads. Besides, the parameters on the two heads are decoupled to avoid interacting with each other. Then, we extend the sensitivity-specific regressor to a cascaded structure. Benefited from the cascaded regressors with increased sensitivity to translation, the performance of our proposed method is significantly improved. The major contributions of this paper are summarized as follows:

- We investigate the availability of highly overlapping RPN proposals for the two basic tasks, which are removed by a Non-Maximum Suppression (NMS) operation. To our best knowledge, few works have been proposed to study whether the highly overlapping proposals are really redundant for both classification and regression. In other words, we study whether the threshold of NMS on RPN proposals is suitable for both tasks.
- We propose a decoupled sampling strategy to select proposals with different densities for the two tasks, corresponding to the sensitivity-specific classifier and regressor. With the decoupled design, it is convenient to extend the single regressor to a cascaded structure with increased sensitivity to translation, in a simple but effective way.
- We evaluate our method from different aspects on the public databases, including PASCAL VOC [23] and Microsoft COCO [24]. The experimental results on MS COCO test-dev demonstrate that our final detector achieves Average Precision (AP) of 45.3 on ResNet-101 and 47.2 on ResNeXt-101-64x4d using multi-scale training.

II. Related Work

In this section, we briefly survey relevant works about the sampling strategy in object detection and the advancements of Faster R-CNN.

A. Sampling Strategy

The training of a deep object detector is essentially an imbalanced foreground-background (fg-bg) class learning problem. Some sampling strategies, such as heuristic sampling or hard negative mining, are performed to maintain a manageable balance between foreground and background. In RPN [12], the authors randomly selected 256 anchor boxes in an image as a mini-batch, where the ratio between the positives and negatives was 1:1. Fast R-CNN [11] constructed a mini-batch by randomly sampling proposals with a fixed fg-bg ratio (1:3). In SSD [8] and OHEM [25], only the gradients of a small set of samples with the largest loss values were back-propagated, which eliminated the hyper-parameter of the fg-bg ratio. In RetinaNet [18], a focal loss was proposed to address the extreme fg-bg class imbalance problem by down-weighting the loss assigned to well-classified examples, and all anchor boxes were contributed to the training. PISA [26] selected the positive samples with the highest IoUs and the negative samples with the highest classification scores to train an object detector, which is defined as prime samples. Besides, some learn-to-match methods such as FreeAnchor [27], ATSS [28] and AutoAssign [29] were proposed to change the contribution of samples.

All the above methods were trained on the same samples, which leads to the interaction between classification and regression.

B. The Advancements of Faster R-CNN

Recently, there are some advancements focused on either classification or regression.

R-FCN [21] introduced positive-sensitive score maps to address the dilemma between invariance/variance on translation for classification and regression tasks. R-FCN-3000 [30] decoupled a super-classes detection by a fine-grained classification. Deformable ConvNets [31], [32] were introduced to adapt to the geometric variations of objects. Cheng *et al.* [33] focused on improving the classification performance of the detector, and proposed a Decoupled Classification Refinement (DCR) module to train an RCNN-styled strong classifier, while introduced extra computation overhead. Furthermore, Cheng *et al.* [34] designed a faster DCR module to alleviate the computation overhead. Cai *et al.* [16] proposed Cascade R-CNN, a multi-stage object detection architecture composed of cascaded regressors and cascaded classifiers, which learned high-quality object detectors sequentially. The cascaded regressors are powerful for bounding box regression and have been evidenced by the remarkable improvement under high-quality evaluation metrics. In order to emphasize the head structure, Double-Head [35] adopted a fully connected head for classification and a convolution head for regression. TSD [36] proposed task-aware spatial disentanglement learning, in which two deformation-learning operations extracted specific features focusing on different spatial dimensions for different tasks. Qiao *et al.* [37] proposed DeFRCN for few-shot object detection, which tailored the degree of decoupling between the three components of Faster R-CNN and decoupled the tasks of classification and regression by a score calibration module. D2Det [38] introduced a dense local regression that predicts multiple dense bounding boxes for each proposal.

Different from the above, our Decoupled R-CNN selects different dense samples for corresponding tasks to satisfy the sensitivity requirements of the two basic tasks.

III. Motivation

A series of detectors based on Faster R-CNN have shown dramatic improvement in object detection, while it is worth noting that the generated RPN proposals highly overlap with each other in the proposal generation stage. To reduce the redundancy, a common practice is to use NMS at an Intersection over Union (IoU)[2] threshold (usually set to 0.7) for the RPN proposals.

The strategy is also proven not harmful to the final performance since the detector is trained on the same proposals for both tasks in [12]. However, there is no previous work studying whether the threshold of NMS is suitable for both classification and regression. To bridge the gap, we next investigate the availability of highly overlapping proposals for specific tasks.

A. The investigation of Highly Overlapping Proposals

To investigate the availability of highly overlapping proposals for the classification and regression, we conduct a toy example with Faster R-CNN on PASCAL VOC [23] dataset. We perform twice sampling on RPN proposals with different NMS thresholds for classification and regression, respectively. Besides, the parameters of Fully Connected (FC) layers from the classification branch and the regression branch are unshared. With such experimental settings, we dynamically change the corresponding NMS thresholds for classification (NMS_cls) and regression (NMS_reg) to explore the impact of the NMS threshold on the two different tasks.

For quantitative analysis of the impact of NMS threshold, we utilize the overall AP[3] as the primary metric. However, the overall metric is not enough to show the specific influence at each single IoU threshold, and we further use (AP_{50}, AP_{60}, AP_{70}, AP_{80}, AP_{90}) to detailedly discuss the influence with a different quality degree. The experimental results are summarized in Table I.

As in Table I, when we increase NMS_reg from 0.7 to 0.9 and fix NMS_cls, the performance is significantly improved, especially for high-quality evaluation metrics. However, the increment of NMS_cls for classification has little influence on the detection performance. The phenomena induce two findings as follows and encourage us to make a distinction when treating classification and regression in a deep detector: 1) highly overlapping proposals are useful rather than redundant for regression, and the regressor trained on denser proposals will fit with translation variance better; 2) highly overlapping proposals are redundant for classification, and [25] also showed that the proposals with high overlaps are likely to have correlated losses, which results in twice repetitive optimization. These observations lead to a question: **Why do the same proposals have different behaviors for the two tasks?**

TABLE I: Results with different NMS thresholds evaluated on the PASCAL VOC2007 test. NMS_reg refers to NMS threshold for regression, and NMS_cls refers to NMS threshold for classification.

NMS_reg	NMS_cls	AP	AP_{50}	AP_{60}	AP_{70}	AP_{80}	AP_{90}
0.7	0.7	54.8	82.2	77.5	67.4	49.2	18.0
0.8	0.7	55.9	82.2	77.9	68.0	51.5	20.7
0.9	0.7	**56.1**	81.9	77.3	67.9	52.5	21.8
0.7	0.8	55.0	81.7	76.7	68.5	49.8	18.5
0.7	0.9	54.8	81.8	77.0	68.0	49.9	17.7

B. Analysis of the Head Branches

To make a comprehensive analysis, we first give concrete notations, followed by our analytical results. Suppose F and G are the corresponding classification function and regression function in R-CNN, respectively. The predictions of the two head branches are given by:

$$p = F(X), \tag{1}$$

$$t = G(X), \tag{2}$$

where X is the RoI feature of a proposal, p is the predicted probability and t is the predicted offset. When training a deep detector, cross-entropy loss and L_1 loss are usually used for classification and regression, respectively. With a concrete declaration, the cross-entropy loss is defined as:

$$CE(p, p^*) = -\sum_i p_i^* \log(p_i), \tag{3}$$

and L_1 loss is defined as:

$$L_1(t_j, t_j^*) = |t_j - t_j^*|, \tag{4}$$

where i is the index of categories, j is the index of a box's center coordinates, and p_i^*, t_j^* are the corresponding ground truth.

For two pre-mentioned highly overlapping proposals, the input features X_1 and X_2 are also highly overlapped such that the predictions of p_1 and p_2 are highly correlated and so it is with the predictions of t_1 and t_2. On the other hand, these two proposals have the same category labels p^*, but different offset labels t_1^* and t_2^* as training supervisions. These two ground-truth offsets (t_1^*, t_2^*) always differ from the shift between the two proposals. Then, under the optimization with the above two object functions for classification and regression, t_1 and

[2]Intersection over union (IoU) between the proposal bounding box b and any ground truth bounding box b_g is defined as:

$$IoU(b, b_g) = \frac{area(b \bigcap b_g)}{area(b \bigcup b_g)}.$$

[3]AP is computed by averaging over 10 IoU thresholds[0.5:0.95]; AP_{50} and others are computed at a single IoU threshold of 0.5 or corresponding value.

t_2 aim to specific targets of t_1^* and t_2^*, while p_1 and p_2 aim to the same target of p^*, leading to different behaviors on the highly overlapping proposals for the two tasks.

C. Analysis of the Effective Bin Locations

The RoI feature of a proposal is generated by RoI-pooling, in which max-pooling is utilized to convert the proposal into a fixed spatial extent of HxW (*e.g.*, 7x7). The visualization of Effective Bin Locations (EBLs) in [32] has shown that not all bins in an RoI-pooling contribute equally to their responses, and bins on the object foreground generally receive larger gradients. For a deep detector, the effective bins are inconsistent for different tasks. Specifically, the features in the salient area have rich information for classification while features around the boundary are always necessary for bounding box regression. Intuitively, the inside bins of an object generally receive larger gradients for classification and the boundary bins of an object receive larger gradients for regression. Thus, the shift of RoIs is unlikely to influence the inside bins. Reconsidering the two highly overlapping proposals, although the RoI features are highly correlated, the corresponding EBLs for the regression task are more likely to be different.

Based on the aforementioned analyses, we find that the regressor has intrinsic characteristics of relative sensitivity to translation while the classifier is relatively insensitive to translation. The phenomena usually happen in the two-stage detectors and encourage us to process the classification and regression with a different strategy: **using the highly overlapping proposals to make the regressor higher sensitive to translation.**

IV. Proposed Method

In this section, we propose Decoupled R-CNN to deal with the sensitivity mismatching for specific tasks, which is illustrated in Fig. 2. We first introduce a decoupled sampling strategy to select different dense proposals, corresponding to the sensitivity-specific classifier and regressor. Further, we extend the single regressor, equipped with increased sensitivity to translation, to cascaded structure in a simple way, which enhances the performance of the detector with a higher accurate localization.

A. Decoupled Sampling Strategy

Sharing the training samples makes the heads of classifier and regressor finally have the same sensitivity to translation, where the two tasks compromise with each other. From the above analyses in Section III, the corresponding object functions of classifier and regressor have different characteristics of translation sensitivity. Therefore, we here design a Decoupled Sampling Strategy (DSS) to separately take samples from specific RPN proposals for preventing the two tasks from interacting with each other. Specifically, we employ NMS with different thresholds (NMS_cls and NMS_reg) on the RPN proposals to obtain the differently dense proposals for the corresponding tasks (Proposals_cls for classification and Proposals_reg for regression), which is shown in Fig. 2. Notice that the regressor is higher sensitive to translation intrinsically than the classifier, NMS_reg threshold should be set higher than NMS_cls. And compared to Proposals_cls, the shift on overlapping Proposals_reg is slighter. Hence, the final proposals of regression should be much more than the proposals of classification.

In this manner, our regressor needs to be aware of the slighter translation between overlapping proposals, obtaining more accurate localization. We set the label attribute by using the function as follows:

$$L = \text{sign}(IoU - p), \tag{5}$$

where

$$\text{sign}(x) = \begin{cases} +1, & x > 0 \\ -1, & x \leq 0 \end{cases} \tag{6}$$

If the IoU ratio is greater than p ($L = +1$), we will assign a positive label to an RPN proposal. Otherwise, we will assign a negative label (without loss of generality, we set $p = 0.5$). To maintain a manageable balance between foreground and background in classification, we randomly select positive and negative proposals with a fixed fg-bg ratio (1:3) from Proposals_cls (Random Sampling with fg-bg Ratio). For regression, we only select positive proposals from Proposals_reg and construct a large mini-batch for more training samples (Positive Sampling).

Our proposed DSS is a focused module, which can deal with different sensitivities to translation for classification and regression. In the previous Faster R-CNN, random sampling with fg-bg ratio for shared proposals causes two limitations for the regressor: the restrictive number of positive and the useless negative; the restrictive sensitivity to translation. In our Decoupled R-CNN, adopting separate sampling only for the positive on more dense Proposals_reg allows us to efficiently train the regressor on more available samples. Training on proposals with different densities for two tasks corresponds with sensitivity-specific heads. Besides, to prevent the two task-specific heads from interacting with each other, the classifier branch should not share any parameters with the regressor branch (the Unshared-FC Heads are shown in Fig. 2).

B. Cascaded Structure for Regression

Cascaded technique is an extremely simple but highly effective way to improve the performance of a deep detector. Considering the decoupled sampling for the two unshared-FC heads, we can extend the single regressor in Decoupled R-CNN to cascaded structure in a simple way, as shown in Fig. 3. Increased translation sensitivity for regression is also suitable for cascaded structure, which further enhances the accurate localization capability of our detector.

Notice that all regression stages have the same architecture, two fully connected layers followed by a final output layer. The positive samples in the first stage ($P\text{-}Samples_1$) are taken from Proposals_reg. In the subsequent stages, we adopt the resampling mechanism

$$P\text{-}Samples_i = Regressor_{i-1}(P\text{-}Samples_{i-1}). \tag{7}$$

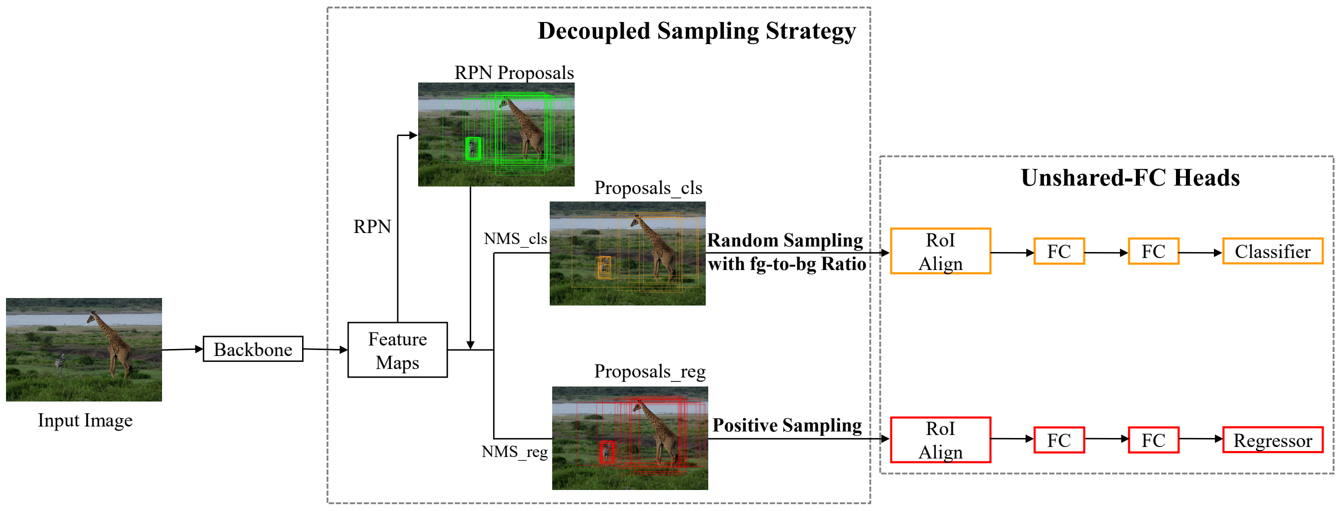

Fig. 2: The framework of Decoupled R-CNN. It decouples the proposals for two tasks (Decoupled Sampling Strategy) and makes the parameters of the following FC layers to be unshared (Unshared-FC Heads). Proposals_cls and Proposals_reg denote the proposals for classification and regression, respectively.

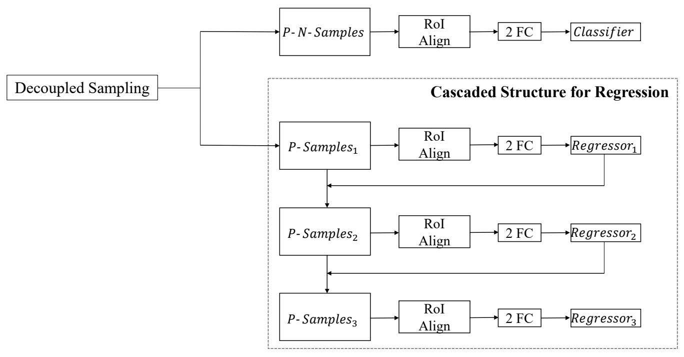

Fig. 3: The two top branches in Decoupled R-CNN with cascaded regression. P-N-$Samples$ denotes the positive and negative samples for classification. P-$Samples_i$ denotes the positive samples for the ith stage of regression.

The regression outputs in the previous stage are sampled to train the subsequent stage. Decoupled sampling strategy selects denser proposals for cascaded regression. Benefiting from it, the cascaded regressors are more sensitive to translation. However, for classification, we do not adopt the cascaded structure because of the little performance improvement, compared with the cascaded regressors, with similar additional computational complexity. The detailed discussion will be introduced in Section VI-A.

C. Loss Function

For the proposed deep detector, we use a multi-task loss to jointly train for classification and cascaded regression:

$$L(p,p^*,t,t^*) = L_{cls}(p,p^*) + \lambda \left[\sum_k L_{loc}(t_k,t_k^*)\right]. \quad (8)$$

The classification loss L_{cls} is the cross-entropy loss defined as $CE(p,p^*) = -\sum_i p_i^* \log(p_i)$, where i is the index of categories, p_i is the predicted probability and p_i^* is the ground-truth label. The regression loss L_{reg} is the L_1 loss defined as $L_1(t_k,t_k^*) = |t_k - t_k^*|$, where k is the index of regression stage, t_k is the predicted offset in k-th stage, and t_k^* is the ground truth. All terms are normalized by mini-batch, and the two task losses are weighted by a balancing parameter λ because of one classifier vs. three cascaded regressors.

V. Experiments

A. Datasets

We evaluate our proposed Decoupled R-CNN on PASCAL VOC [23] and MS COCO [24], which are the two most widely used datasets in object detection. For the PASCAL VOC dataset, we use the union of VOC 2007 trainval and VOC 2012 trainval as training data and evaluate our model on the VOC 2007 test. For the MS COCO dataset, training and evaluation are performed on 118k images of the training set and 5k images of the validation set, respectively. Finally, we report our results on the COCO test-dev set (without public labels).

B. Implementation Details

All experiments are implemented on MMDetection [39], an open-source object detection toolbox based on PyTorch [40]. The input images are resized to 800 pixels along the shorter side. Besides, only the horizontal image flipping is used for data augmentation. The proposed model is end-to-end trained on 4 TITAN X GPUs with a total batch size of 8 (2 images per GPU). For the classification branch, we select 512 region proposals including foreground and background. For the (cascaded) regression branch, we use a large mini-batch size of 512 to only select positive proposals. If the number of positive proposals is less than 512, we pad the negative proposals to ensure a fixed mini-batch size, and the negatives are ignored via setting the corresponding weights to zero.

To train the model, we choose the SGD optimizer with the weight decay and momentum setting as 0.0001 and 0.9, respectively. On the PASCAL VOC, we use a training schedule with 12 epochs, in which the learning rate is initialized as

0.005 and dropped 10 times at the 9th epoch. On the MS COCO, two training schedules are adopted: "1x" and "2x", with corresponding 12 epochs and 24 epochs, respectively. The learning rate is initialized as 0.01 and dropped 10 times at the 8th and 11th epoch in "1x" schedule. For "2x", the learning rate is also initialized as 0.01 while dropped at the 16th and 22nd epoch. For testing, we use at most 1k top-scoring proposals for detection, followed by NMS with a threshold of 0.5, and finally the top-N scoring detections are obtained. Note that before NMS a threshold of score_thr is usually used to remove detections with scores lower than it. Unless otherwise claimed, we set score_thr as 0.05 following the default hyper-parameter in MMDetection.

C. Ablation Experiments

In Decoupled R-CNN, the choice of hyper-parameters (NMS_reg, Number of stages, λ) heavily influences the detection performance. To investigate the effectiveness of these hyper-parameters, we firstly conduct ablation experiments on PASCAL VOC data set and choose the best one fixed in the following experiments. we use ResNet-101 [4] as backbone model with Feature Pyramid Network (FPN) [41] in our ablation experiments.

1) The Necessity of Unshared Parameters: DSS (Decoupled Sampling Strategy) is a key sampling technique for our method, which makes the classifier head and regressor head have different sensitivities to the translation. We set NMS_reg $= 0.7$ to reveal the effect of the different Shared-FC and Unshared-FC head branches for the detection performance, which is reported in Table II.

TABLE II: The ablation study on the shared parameters of FC layers.

Method	**Shared-FC**	**Unshared-FC**	**AP**
FPN	✓		54.1
		✓	55.0
Deocupled R-CNN	✓		53.6
		✓	55.0

From Table II, whatever FPN or Decoupled R-CNN, the models with unshared-FC have achieved the best performance. Especially, Decoupled R-CNN with shared-FC has degraded the performance. That means that sharing a same fully connected head for two different tasks would introduce confliction. Hence, unshared parameters are necessary for the proposed method.

2) NMS_reg Threshold: By using DSS, the classification and the regression have differently dense samples. We explore different choices of NMS_reg threshold for regression to investigate the influence of NMS_reg Threshold.

The detection performance with different NMS_reg, which varies from 0.7 to 0.95, are reported in Table III. Obviously, increasing the NMS_reg value will improve the performance, and it achieves the best performance of 56.6 AP when NMS_reg is equal to 0.85. However, the performance degrades when we further increase the value of NMS_reg. The phenomena seem that the translation would be too slight to produce a response for regression.

TABLE III: NMS_reg threshold changes the sensitivity of the regressor head in Decoupled R-CNN.

Method	**NMS_reg**	**AP**	**AP_{50}**	**AP_{60}**	**AP_{70}**	**AP_{80}**	**AP_{90}**
Decoupled R-CNN	0.7	55.0	82.2	77.5	68.3	49.8	18.2
	0.75	55.3	81.8	78.0	68.1	50.2	19.4
	0.8	56.0	82.2	78.0	67.9	52.0	21.5
	0.85	**56.6**	81.9	77.9	68.8	53.1	21.9
	0.9	56.5	81.6	77.3	68.6	53.0	23.1
	0.95	56.1	81.5	77.5	68.2	52.4	22.6

Hence, based on the availability of highly overlapping proposals, we introduce DSS to select proposals with different densities for the two tasks, corresponding to sensitivity-specific heads. The regressor, equipped with specific sensitivity to translation, is capable to achieve higher accurate localization.

3) Number of Stages: With the additional cascaded structure in our proposed model, we next explore the impact of the number of stages for cascaded regression on the detection performance, and we exhibit results in Table IV.

TABLE IV: The performance of cascaded regression.

stages	**AP**	**AP_{50}**	**AP_{60}**	**AP_{70}**	**AP_{80}**	**AP_{90}**
1	56.6	81.9	77.9	68.8	53.1	21.9
2	58.4	81.0	77.1	69.1	56.0	29.3
3	**58.7**	81.2	77.3	69.2	55.9	30.2
4	58.5	81.1	76.8	69.1	55.7	30.8

Similar to Cascade R-CNN [16], the three-stage regression achieves the best result. As the number of stages increases to 4, the performance increases continuously on the AP_{90} metric but declines slightly on AP_{50}, which means that the aimlessness of increasing the number of stages would result in loss imbalance for the two tasks due to cascaded regressors vs. one classifier.

4) The Impact of NMS_reg for Cascaded Regression: Compared with Cascade R-CNN [16], the training samples for our cascaded regressors are denser through DSS. As shown in Table V, the cascaded regressors trained on denser proposals (NMS_reg=0.85), improves the AP by 0.7 , which reflects the impact of NMS_reg for cascaded regression. The improvement indicates that increased sensitivity to translation for the regressor head is also necessary for the cascaded structure.

TABLE V: The impact of NMS_reg for cascaded regression.

NMS_reg	**AP**	**AP_{50}**	**AP_{60}**	**AP_{70}**	**AP_{80}**	**AP_{90}**
0.7	58.0	81.5	77.8	69.4	54.9	27.0
0.85	58.7	81.2	77.3	69.2	55.9	30.2

5) Balance Between Classification and Regression: The hyper-parameter of λ in Eq. (8) controls the balance between the two tasks. When using a single regressor, although we select more dense positive proposals, the gap between the two terms of loss is not large. In this case, we empirically set $\lambda = 1$. When using the cascaded structure, the gap is non-negligible, due to three cascaded regressors vs. one classifier.

So it is necessary to select the suitable λ to balance the two task losses. Next, we investigate the influence of λ for the performance when it varies in the range $[0.25, 1]$.

TABLE VI: The influence of λ.

λ	AP	AP_{50}	AP_{60}	AP_{70}	AP_{80}	AP_{90}
1	58.7	81.2	77.1	69.0	56.6	30.0
0.75	58.6	81.6	77.2	69.2	56.0	30.0
0.5	59.0	82.2	77.8	70.8	56.1	29.4
0.25	58.0	82.1	77.8	69.5	54.4	27.5

As shown in Table VI, when decreasing λ from 1 to 0.5, we observe the best overall performance when it is set to 0.5, where our model consistently achieves the best performance, especially on AP_{50}, which is important for classification. The phenomena indicate that there exists an imbalance between the classification loss and the cascaded regression loss. While the performance on high IoU levels significantly degrades as we further decrease λ to 0.25. Therefore, we set $\lambda = 0.5$ in the following experiments.

D. Main Results

1) Results on VOC: First, We evaluate our method on PASCAL VOC following the above settings, and the detection results are reported in Table VII. Compared with FPN based on ResNet-50, our method significantly improves the performance of AP by 2.6, and our cascaded regression further improves the AP by 2.9. Besides, a significant improvement is also shown in the ResNet-101 experiment. Apart from the detection performance, the Inference Speed has also been compared. For Decoupled R-CNN without cascaded regression, the increase in computing overhead is negligible compared to the performance gain. Therefore, in order to achieve better detection results, the further use of cascade technology is also worthwhile.

2) Results on COCO validation: To prove the effectiveness of our method, we report our results on COCO validation based on different backbones. We adopt the "1x" training schedule for the following experiments. The performance on three popular backbones has been reported in Table VIII.

We find that Decoupled R-CNN has improved on these backbones consistently by $1.1 \sim 1.2$ for AP, which demonstrates the effectiveness of our proposed DSS design. When using the cascaded technique for the basic model (Decoupled R-CNN), a consistent improvement with $1.4 \sim 1.5$ AP has been achieved. Especially, our method gains $7 \sim 8$ improvement on AP_{90}, compared to FPN. The quantification results show that our method is effective for improving the performance of the detector with a highly accurate localization. That means the sensitivity-specific heads and cascaded structure in our method have proven to be useful. Besides, from Table VIII, the proposed module only brings about few FLOPs (floating point operations) and parameters, which is tolerable in detection.

To confirm the high accurate localization, we further visually show some results of the head-to-head comparison between Faster R-CNN [12] and Ours in Fig. 4. Clearly, the localization of our output regression boxes achieves higher accuracy.

3) Results on COCO test-dev: In this part, We will carry out a comprehensive and systematic analysis about our proposed method compared with a series of state-of-the-art methods on COCO test-dev. For a fair comparison, the results of single-model and single-scale testing for all methods are reported in Tabel IX [4].

First, when compared with the one-stage detectors, whose results located in the top group of Table IX, our proposed method has achieved a comparable performance. It is noted that the recent one-stage detectors usually adopt a multi-scale training manner for improving the performance. For a fair comparison, we further adapt multi-scale training for our model, and our detector (our method with ResNet-101 and MS_{train}) achieves a significantly better performance of 45.3 AP, which outperforms existing one-stage approaches.

Compared with the two-stage detectors, whose results located in the middle group of Table IX, our detector with ResNet-101 using single-scale training achieves 43.3 AP, which outperforms the recent two-stage approaches of TridentNet [43], Cascade R-CNN [16] and TSD [36]. However, DCNv2 [32] and D2Det [38] have a better performance than ours. Notice that DCNv2 uses the deformable convolution, where it has proven the deformable convolution is better than the normal one, to achieve 44.0 AP on ResNet-101. However, the gaps reduce to 0.1 (44.6 vs. 44.5) when comparing on the backbone of ResNeXt-101. Besides, D2Det proposes a dense local regression and a discriminative RoI pooling for classification, achieving 45.4 AP. Considering the improvement from the proposed RoI pooling (more than 1.0 AP), the difference between D2Det and our method ($Ours^{+}$)[5] is less than 0.7 when only comparing the localization capability.

We further improve the detection performance of our method to 46.3 and 47.2 by utilizing larger backbones ResNeXt-101-32x4d and ResNeXt-101-64x4d, respectively. The results show that our proposed DSS is effective for improving the detection performance.

For further demonstrating the performance of our proposed model, we select visual results of compared methods on the COCO test-dev and show them in Fig. 5. Our model accurately locates the boundary of objects and achieves better results under challenging conditions, such as different geometric variations in object scale, pose, viewpoint, etc. However, there also have some failure cases as in the last row, where the objects are partially occluded or aim clutter. With considerable observation, we find that foreground objects highly overlap with each other in these scenarios, therefore confusing our regressor and leading to the final inaccurate bounding boxes. In the future, we will explore how to solve these conditions.

VI. Discussion

To detailedly explain the advantages of our Decoupled R-CNN, we next discuss the differences between it and other state-of-the-art models from the following three aspects.

[4]We use the "2x" training schedule and set the score_thr to 0.001.

[5]$Ours^{+}$ means that we apply the same soft-NMS as D2Det at inference and obtain 43.7 AP.

TABLE VII: Detection results on PASCAL VOC. Inference speed is measured on a single TITAN X GPU.

Backbone	Method	Cascaded Regression	Inference Speed (sec./image)	AP	AP$_{50}$	AP$_{60}$	AP$_{70}$	AP$_{80}$	AP$_{90}$
ResNet-50	FPN	✗	0.079	50.6	80.2	75.4	63.7	42.3	11.5
	Decoupled R-CNN	✗	0.087	53.2	80.1	75.5	65.8	47.3	17.7
	Decoupled R-CNN	✓	0.133	56.1	80.9	76.4	67.6	52.1	24.2
ResNet-101	FPN	✗	0.101	54.1	82.1	77.9	67.4	48.5	15.6
	Decoupled R-CNN	✗	0.108	56.6	81.9	77.9	68.8	53.1	21.9
	Decoupled R-CNN	✓	0.154	59.0	82.2	77.8	70.8	56.1	29.4

TABLE VIII: Detailed comparison on COCO validation over multiple popular backbones. Inference speed is measured on a single TITAN X GPU and FLOPs are computed on a given size image of 1333x800.

Backbone	Method	Cascaded Regression	Inference Speed (sec./image)	FLOPs (GFLOPs)	Model Parameters (M)	AP	AP$_{50}$	AP$_{60}$	AP$_{70}$	AP$_{80}$	AP$_{90}$
ResNet-50	FPN	✗	0.117	215.82	41.53	37.4	58.1	53.6	45.9	33.6	11.1
	Decoupled R-CNN	✗	0.125	229.72	55.42	38.6	58.6	54.3	47.1	34.9	13.8
	Decoupled R-CNN	✓	0.164	257.52	83.22	40.1	58.2	54.0	47.7	37.3	19.0
ResNet-101	FPN	✗	0.156	295.70	60.52	39.4	60.1	55.6	48.3	35.6	12.9
	Decoupled R-CNN	✗	0.159	309.59	74.42	40.6	60.5	56.2	49.4	36.7	16.0
	Decoupled R-CNN	✓	0.200	337.39	102.22	42.1	60.0	55.8	49.6	39.6	20.9
ResNeXt-101-32x4d [42]	FPN	✗	0.185	299.61	60.16	41.2	62.2	57.8	50.6	37.8	14.6
	Decoupled R-CNN	✗	0.192	313.51	74.05	42.3	62.4	58.2	51.1	38.9	17.1
	Decoupled R-CNN	✓	0.238	341.31	101.85	43.7	62.2	58.1	51.7	40.9	21.9

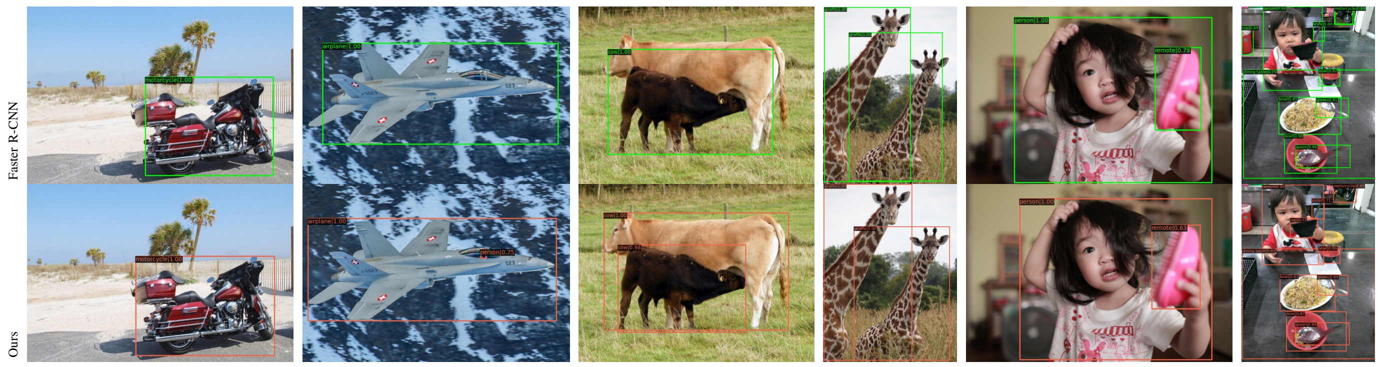

Fig. 4: Faster R-CNN [12] (top) vs. Decoupled R-CNN (bottom). Both are using ResNet-101 with FPN as the backbone. It is clear that our rectangle boxes more tightly bound the objects than the Faster R-CNN.

A. *Differences to Cascade R-CNN*

Our module of cascaded regression is similar to Cascade R-CNN. However, the proposed work differs from it in that:

1) Cascade R-CNN is composed of cascaded regressors and cascaded classifiers, averaging the three classifier probabilities for the final classification score at inference. However, our proposed method only has cascaded regressors with a single classifier. This design profits from the study [53] that the classification score does not well reflect the quality of the bounding box (localization). It has indicated that classification and localization need to be solved differently in the detection pipeline. Specifically, given a proposal, the classifier probability naturally acts as classification confidence of the proposal, and the bounding box regression finds the optimal transformation for the proposal to best fit the ground-truth. Therefore, the classification score is not well correlated with the localization confidence. Via the considerably designed decoupling technique, the structure of our method is feasible and the classification score depends on the single classifier probability on the RPN proposals. Moreover, considering our proposed DSS module and Unshared-FC heads, it is convenient to only adopt cascaded regression without cascaded classification.
2) For training a sequence of higher-quality detectors to effectively reject close false positives, Cascade R-CNN increases IoU thresholds at each cascade stage for selecting the positive samples with higher IoU distributions. However, in our cascaded regression, the resampling is only performed on the positive, and the output IoU of a regressor is almost invariably better than the input IoU.

TABLE IX: Comparisons with state-of-the-art detectors on the COCO test-dev. "MS$_{train}$" denotes multi-scale training. "n/a" means that trained models from the MMDetection repository and AP results from the original papers are not available. "ResNeXt-101(32x4d)" denotes the backbone model that we measure the inference speed, FLOPs, and model parameters on. Inference speed, FLOPs, and model parameters are measured on the same machine with a single TITAN X GPU under the same MMDetection [39] framework. Especially, inference speed is related to score_thr, for which we remove detections with scores lower than it, and set score_thr to 0.001; FLOPs is highly related to the input shape, which we set to [1333, 800]. "Ours" represents single-scale training + traditional NMS; "Ours$^+$" represents single-scale training + soft-NMS; and "Ours*" represents multi-scale training + soft-NMS.

Method	Backbone	MS$_{train}$	Inference Speed (sec./image)	FLOPs (GFLOPs)	Model Parameters (M)	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
one-stage detectors											
DES [44]	VGG16		n/a	n/a	n/a	32.8	53.2	34.6	13.9	36.0	47.6
RetinaNet [18]	ResNet-101		0.149	250.34	37.74	39.1	59.1	42.3	21.8	42.7	50.2
CornerNet [45]	Hourglass-104		0.435	1848.93	201.04	40.5	56.5	43.1	19.4	42.7	53.9
HSD [46]	ResNet-101		n/a	n/a	n/a	40.2	59.4	44.0	20.0	44.4	54.9
HSD [46]	ResNeXt-101		n/a	n/a	n/a	41.9	61.1	46.2	21.8	46.6	57.0
FoveaBox [47]	ResNet-101	✓	0.154	331.51	57.38	40.8	61.4	44.0	24.1	45.3	53.2
FSAF [48]	ResNet-101	✓	0.133	295.67	55.19	40.9	61.5	44.0	24.0	44.2	51.3
FSAF [48]	ResNeXt-101-64x4d	✓	0.263	460.51	93.92	42.9	63.8	46.3	26.6	46.2	52.7
FCOS [49]	ResNet-101	✓	0.128	289.68	50.96	41.5	60.7	45.0	24.4	44.8	51.6
FCOS [49]	ResNeXt-101-64x4d	✓	0.263	459.48	89.79	44.7	64.1	48.4	27.6	47.5	55.6
FreeAnchor [27]	ResNet-101	✓	0.154	330.21	56.74	43.1	62.2	46.4	24.5	46.1	54.8
FreeAnchor [27]	ResNeXt-101-64x4d	✓	0.303	495.05	95.46	44.9	64.3	48.5	26.8	48.3	55.9
ATSS [28]	ResNet-101	✓	0.135	294.64	51.06	43.6	62.1	47.4	26.1	47.0	53.6
ATSS [28]	ResNeXt-101-64x4d	✓	0.263	460.54	89.79	45.6	64.9	49.7	28.5	48.9	55.6
AutoAssign [29]	ResNet-101	✓	0.169	290.85	55.09	44.5	64.3	48.4	25.9	47.4	55.0
AutoAssign [29]	ResNeXt-101-64x4d	✓	0.303	460.65	93.92	46.5	66.5	50.7	28.3	49.7	56.6
two-stage detectors											
Faster R-CNN w/FPN [41]	ResNet-101		0.149	295.7	60.52	36.2	59.1	39.0	18.2	39.0	48.2
Mask R-CNN [13]	ResNeXt-101(32x4d)		0.208	352.69	62.80	39.8	62.3	43.4	22.1	43.2	51.2
Libra R-CNN [50]	ResNet-101		0.156	296.8	60.78	41.1	62.1	44.7	23.4	43.7	52.5
Libra R-CNN [50]	ResNeXt-101-64x4d		0.284	461.64	99.51	43.0	64.0	47.0	25.3	45.6	54.6
Faster R-CNN w/ PISA [26]	ResNeXt-101(32x4d)		0.204	299.61	60.16	42.3	62.9	46.8	24.8	45.5	53.1
Grid R-CNN [51]	ResNet-101		0.172	408.86	83.31	41.5	60.9	44.5	23.3	44.9	53.1
Grid R-CNN [51]	ResNeXt-101(32x4d)		0.208	412.77	82.95	43.2	63.0	46.6	25.1	46.5	55.2
Double-Head-Ext [52]	ResNet-101		0.294	569.9	66.11	42.3	62.8	46.3	23.9	44.9	54.3
TridentNet [43]	ResNet-101		0.256	864.85	52.51	42.7	63.6	46.5	23.9	46.6	56.6
TridentNet [43]	ResNet-101-DCN	✓	n/a	n/a	n/a	46.8	67.6	51.5	28.0	51.2	60.5
Cascade R-CNN [16]	ResNet-101		0.196	323.34	88.16	42.8	62.1	46.3	23.7	45.5	55.2
TSD [36]	ResNet-101		0.222	312.29	98.86	43.2	64.0	46.9	24.0	46.3	55.8
DCNv2 [32]	ResNet-101		0.185	227.09	61.81	44.0	65.9	48.1	23.2	47.7	59.6
DCNv2 [32]	ResNeXt-101(32x4d)		0.244	301.64	62.73	44.6	n/a	n/a	n/a	n/a	n/a
D2Det [38]	ResNet-101		0.250	379.29	98.86	45.4	64.0	49.5	25.8	48.7	58.1
Ours	ResNet-101		0.200	337.39	102.22	43.3	62.0	46.4	24.1	46.3	55.3
Ours$^+$	ResNet-101		0.200	337.39	102.22	43.7	61.9	47.4	24.4	46.8	55.8
Ours*	ResNet-101	✓	0.200	337.39	102.22	45.3	63.4	49.1	26.5	48.4	56.5
Ours	ResNeXt-101-32x4d		0.238	341.31	101.85	44.5	63.5	47.8	25.4	47.3	56.2
Ours*	ResNeXt-101-32x4d	✓	0.238	341.31	101.85	46.3	64.7	50.4	27.8	49.4	57.5
Ours*	ResNeXt-101-64x4d	✓	0.323	502.23	140.94	47.2	65.6	51.2	29.0	50.5	58.9

It makes us use a fixed IoU threshold of 0.5 for all the cascaded regression stages. Our method eliminates the hyperparameters of increasing IoU thresholds, and each stage of regressors is trained on the corresponding sample distributions.

3) The most important difference is training samples. Cascade R-CNN uses random sampling with fg-bg ratio to select shared proposals, which inevitably restricts the regressor's sensitivity to translation. Without the constraint of classification, we select more dense positive proposals to train the cascaded regressor. Benefited from increased translation sensitivity, our cascaded regressors can localize an object more accurately.

To concretely compare with Cascade R-CNN, we implement our Decoupled R-CNN with the same cascaded structure and the increasing IoU thresholds. Table X summarizes the stage performance over the two different methods. And we can find that the regressor at the final third stage brings significant

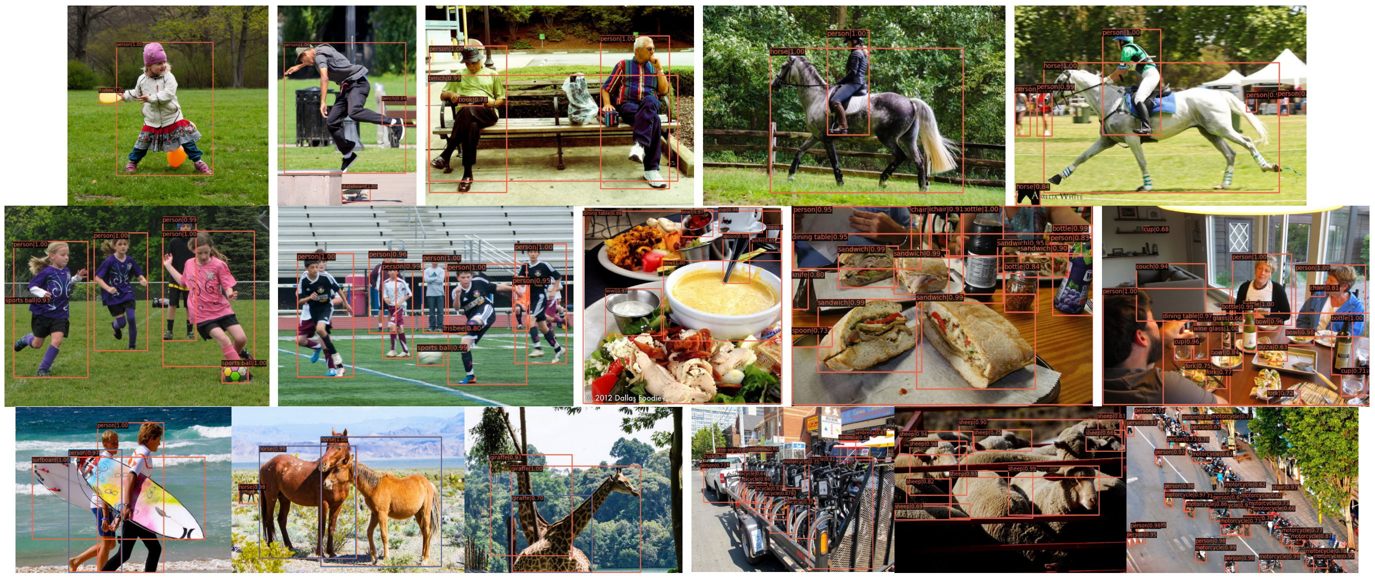

Fig. 5: Visualization on the COCO test-dev set. Each bounding box is linked with a category label and a softmax score. We set the score threshold to 0.6 to display these images. The last row shows some failure cases, where the objects are partially occluded or aim clutter in the challenging non-iconic images. In these cases, different foreground objects overlap with each other, therefore confusing our regressor and leading to inaccurate bounding boxes.

TABLE X: The stage performance of cascaded regression and cascaded classification over different methods. Note that we equip Decoupled R-CNN with both cascaded regression and cascaded classification, in which we adopt the increasing IoU thresholds at the cascaded stages as same as Cascade R-CNN. And both methods use ResNet-101 as the backbone. The test stage (i, j) denotes the i-th stage regressor and the j-th stage classifier, and the test stage (3, 1-3) denotes the average of the three classifier probabilities.

Method	**Inference Speed**	**FLOPs**	**Model Parameters**	**Test Stage**					
	(sec./image)	(GFLOPs)	(M)	(1, 1)	(2, 1)	(3, 1)	(3, 2)	(3, 3)	(3, 1-3)
Cascade R-CNN	0.196	323.34	88.16	38.5	40.5	41.2	41.9	41.5	42.0
Decoupled R-CNN	0.222	365.03	129.85	38.7	41.4	41.8	42.1	41.5	42.7

improvements for Cascaded R-CNN (from 38.5 to 41.2) and Decoupled R-CNN (from 38.7 to 41.8). However, the single classifier at any stage obtains marginal improvements and the ensemble result, averaging the three classifier probabilities, is necessary for better performance. Benefiting from our proposed DSS, which is more sensitive to translation, our cascaded regressors are better than the regressors in cascade R-CNN (41.8 vs. 41.2). When adopting the cascaded classification, we obtain similar performance improvements with similar additional computational complexity. So our final detector only adopts the cascaded structure for regression.

B. Sampling Strategy

Recently, some learn-to-match approaches are proposed to change the contribution of samples. FreeAnchor [27], an object-anchor matching approach, is proposed to allow each object to flexibly match the best anchors during training. ATSS [28] uses the statistical characteristics of each object to compute a dynamic IoU threshold for the definition of positive and negative samples. AutoAssign [29] generates positive and negative weight maps to adjust each location's positive and negative confidences. However, they neglect that the same sample has different contributions to the two tasks. For our proposed DSS, "classification and regression" take samples from the different dense RPN-proposals, and our model evenly deals with all samples in the training process.

In TSD [36], two disentangled proposals are generated by two deformable operations on the original proposals and the corresponding feature extractors for the two heads, solving the spatial misalignment between classification and regression. During training, the TSD heads need to be jointly optimized with the sibling heads. At inference, the same deformable operations and feature extractors in TSD heads are applied. But our proposed method focuses on different sensitivity to the translation for classification and regression, where DSS takes different dense samples for the unshared-FC heads of the classifier and the regressor. Even equipped with cascaded regression, our method has a faster inference speed than the complex TSD heads, which as shown in Table IX.

Besides, SPFTN [54] incorporates a similar strategy and decouples the involved self-paced regularizers for different tasks of localization and segmentation. However, the work focuses on the task of weakly labeled video object localization and segmentation, which is very different from our decoupling of the sampling strategy for the two basic tasks of classification and regression.

C. Regression Task

In Decoupled R-CNN, we decouple the sampling of classification and regression, and disentangle the shared parameters of these two detector heads. In Double-Head [52], a more elaborate structure of the detector heads is designed, obtaining a satisfactory performance. Inspired by the success, we further inspect the effectiveness of Double-Head for our Decoupled R-CNN in Table XI. To implement the Double-Head structure, we equip the classifier head with two fully connected layers and additionally stack 4 residual convolution blocks for the regressor head. With such a replacement operation, Decoupled R-CNN without cascaded regression achieves 40.0 AP and 41.6 AP on ResNet-50 and ResNet-101, which is close to the results with cascaded regression. Furthermore, when our cascaded regression also adopts the 4 residual convolution blocks, we obtain 40.4 AP and 42.2 AP on ResNet-50 and ResNet-101, only a slight gain by $0.1 \sim 0.3$ AP. Although Double-Head has fewer model parameters, there is a great increase in computational overhead. Similarly, the dense local regression in D2Det [38] also adopts a deep head with eight convolutions, which inevitably increases the FLOPs as seen in Table IX. Compared with a deep and complex regressor, our cascaded regressors decompose the regression task into a sequence of stages, which is efficient and low cost.

VII. Conclusion

In this work, we propose a decoupled architecture, called Decoupled R-CNN, for object detection. It is a simple but accurate and efficient method to address a rarely explored problem, i.e. the compromising suboptimality of sensitivity to translation for the current classifier and regressor. Our model uses decoupled sampling strategy to decouple the proposals sampling for two different branches, which guarantees the consistent sensitivity to the translation for each corresponding branch. Furthermore, we propose a cascaded structure for regression to improve the accuracy of localization of Decoupled R-CNN. The extensive experimental results show that our method achieves competitive performance. Inspired by the learn-to-match, we further conjecture that a task-proposal matching approach allows each object to respectively match the best proposals for the two tasks "classification and regression" and the label assignment for the two tasks could be conducted in a different manner, which encourages us to further improve our model in the future.

VIII. Acknowledgement

This work was supported by the National Natural Science Foundation of China under Grant No. 12001180, 62071327 and 61801325.

References

[1] S. Hou, C. Zhao, Z. Chen, J. Wu, Z. Wei, and D. Miao, "Improved instance discrimination and feature compactness for end-to-end person search," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2021.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[6] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," pp. 6517–6525, 2017.

[7] ——, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

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

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[11] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[12] 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*, 2015, pp. 91–99.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[14] J. Huang, Z. Chen, Q. M. Jonathan Wu, C. Liu, H. Yuan, and W. He, "Catfpn: Adaptive feature pyramid with scale-wise concatenation and self-attention," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2021.

[15] H. Wang, Q. Wang, H. Zhang, J. Yang, and W. Zuo, "Constrained online cut-paste for object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2020.

[16] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6154–6162.

[17] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Single-shot refinement neural network for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4203–4212.

[18] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[19] Z. Shen, Z. Liu, J. Li, Y.-G. Jiang, Y. Chen, and X. Xue, "Dsod: Learning deeply supervised object detectors from scratch," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1919–1927.

[20] R. Zhu, S. Zhang, X. Wang, L. Wen, H. Shi, L. Bo, and T. Mei, "Scratchdet: Training single-shot object detectors from scratch," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2268–2277.

[21] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Advances in neural information processing systems*, 2016, pp. 379–387.

[22] B. Singh and L. S. Davis, "An analysis of scale invariance in object detection snip," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3578–3587.

[23] 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*, vol. 88, no. 2, pp. 303–338, 2010.

[24] 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*. Springer, 2014, pp. 740–755.

[25] 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*, 2016, pp. 761–769.

TABLE XI: The effectiveness of Double-Head for Decoupled R-CNN.

Method	Backbone	Cascaded Regression	Double-Head	Inference Speed (sec./image)	FLOPs (GFLOPs)	Model Parameters (M)	AP
Decoupled R-CNN	ResNet-50	✗	✗	0.125	229.72	55.42	38.6
		✗	✓	0.270	490.02	47.12	40.0
		✓	✗	0.164	257.52	83.22	40.1
		✓	✓	0.556	1038.83	58.30	40.4
	ResNet-101	✗	✗	0.159	309.59	74.42	40.6
		✗	✓	0.313	569.90	66.11	41.6
		✓	✗	0.200	337.39	102.22	42.1
		✓	✓	0.625	1118.21	77.29	42.2

[26] Y. Cao, K. Chen, C. C. Loy, and D. Lin, "Prime sample attention in object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 583–11 591.

[27] X. Zhang, F. Wan, C. Liu, R. Ji, and Q. Ye, "FreeAnchor: Learning to match anchors for visual object detection," in *Neural Information Processing Systems*, 2019.

[28] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9759–9768.

[29] B. Zhu, J. Wang, Z. Jiang, F. Zong, S. Liu, Z. Li, and J. Sun, "Autoassign: Differentiable label assignment for dense object detection," *arXiv preprint arXiv:2007.03496*, 2020.

[30] B. Singh, H. Li, A. Sharma, and L. S. Davis, "R-fcn-3000 at 30fps: Decoupling detection and classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1081–1090.

[31] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 764–773.

[32] X. Zhu, H. Hu, S. Lin, and J. Dai, "Deformable convnets v2: More deformable, better results," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9308–9316.

[33] B. Cheng, Y. Wei, H. Shi, R. Feris, J. Xiong, and T. Huang, "Revisiting rcnn: On awakening the classification power of faster rcnn," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 453–468.

[34] ——, "Decoupled classification refinement: Hard false positive suppression for object detection," *arXiv preprint arXiv:1810.04002*, 2018.

[35] Y. Wu, Y. Chen, L. Yuan, Z. Liu, L. Wang, H. Li, and Y. Fu, "Double-head rcnn: Rethinking classification and localization for object detection," *arXiv preprint arXiv:1904.06493*, vol. 2, 2019.

[36] G. Song, Y. Liu, and X. Wang, "Revisiting the sibling head in object detector," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 563–11 572.

[37] L. Qiao, Y. Zhao, Z. Li, X. Qiu, J. Wu, and C. Zhang, "Defrcn: Decoupled faster r-cnn for few-shot object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 8681–8690.

[38] J. Cao, H. Cholakkal, R. M. Anwer, F. S. Khan, Y. Pang, and L. Shao, "D2det: Towards high quality object detection and instance segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 485–11 494.

[39] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "MMDetection: Open mmlab detection toolbox and benchmark," *arXiv preprint arXiv:1906.07155*, 2019.

[40] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, pp. 8026–8037, 2019.

[41] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2117–2125.

[42] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500.

[43] Y. Li, Y. Chen, N. Wang, and Z. Zhang, "Scale-aware trident networks for object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6054–6063.

[44] Z. Zhang, S. Qiao, C. Xie, W. Shen, B. Wang, and A. L. Yuille, "Single-shot object detection with enriched semantics," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5813–5821.

[45] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *Proceedings of the European Confredmon2016youerence on Computer Vision (ECCV)*, 2018, pp. 734–750.

[46] J. Cao, Y. Pang, J. Han, and X. Li, "Hierarchical shot detector," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9705–9714.

[47] T. Kong, F. Sun, H. Liu, Y. Jiang, L. Li, and J. Shi, "Foveabox: Beyound anchor-based object detection," *IEEE Transactions on Image Processing*, vol. 29, pp. 7389–7398, 2020.

[48] C. Zhu, Y. He, and M. Savvides, "Feature selective anchor-free module for single-shot object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 840–849.

[49] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *Proceedings of the IEEE international conference on computer vision*, 2019, pp. 9627–9636.

[50] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra r-cnn: Towards balanced learning for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019, pp. 821–830.

[51] X. Lu, B. Li, Y. Yue, Q. Li, and J. Yan, "Grid r-cnn," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7363–7372.

[52] Y. Wu, Y. Chen, L. Yuan, Z. Liu, L. Wang, H. Li, and Y. Fu, "Rethinking classification and localization for object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 186–10 195.

[53] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang, "Acquisition of localization confidence for accurate object detection," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 784–799.

[54] D. Zhang, J. Han, L. Yang, and D. Xu, "Spftn: A joint learning framework for localizing and segmenting objects in weakly labeled videos," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 2, pp. 475–489, 2020.

Dong Wang received the B.S. degree from Yanbian University, in 2014. He is a PhD candidate in Center for Applied Mathematics, Tianjin University. His current research interests are deep learning for object detection.

Kun Shang received the B.S. degree from the Faculty of mathematics and statistics, Hubei University, in 2011. In 2018, He received the M.S. and Ph.D. degrees from Center for Applied Mathematics, Tianjin University. From 2018 to 2021, he was a assistant professor of Mathematics with the School of Mathematics, Hunan University. In 2021, he joined Research Center for Medical AI, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences. Recently, his research interests include medical image reconstruction and analysis, pattern recognition, compressed sensing, image processing, low-rank tensor minimization and optimization theory and algorithm etc.

Huaming Wu received the B.S. and M.S. degrees from Harbin Institute of Technology, China in 2009 and 2011, respectively, both in electrical engineering. He received the Ph.D. degree with the highest honor in computer science at Freie Universitäat Berlin, Germany in 2015. He is currently an associate professor in the Center for Applied Mathematics, Tianjin University. His research interests include model-based evaluation, wireless and mobile network systems, mobile cloud computing and deep learning.

Ce Wang received the B.S. degree from the Department of Mathematics, Jilin University, in 2015 and Ph.D. degree from the Center for Combinatorics, Nankai University in 2020. He is currently a postdoctoral researcher in the Institute of Computing Technology, CAS. His main interests include generative modeling, image processing, medical image reconstruction and analysis.