Dynamic R-CNN: Towards High Quality Object Detection via Dynamic Training

Hongkai Zhang[1,2], Hong Chang[1,2], Bingpeng Ma[2],
Naiyan Wang[3], and Xilin Chen[1,2]

[1] Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, Chinese Academy of Sciences
[2] University of Chinese Academy of Sciences, [3] TuSimple
hongkai.zhang@vipl.ict.ac.cn, changhong@ict.ac.cn, bpma@ucas.ac.cn,
winsty@gmail.com, xlchen@ict.ac.cn

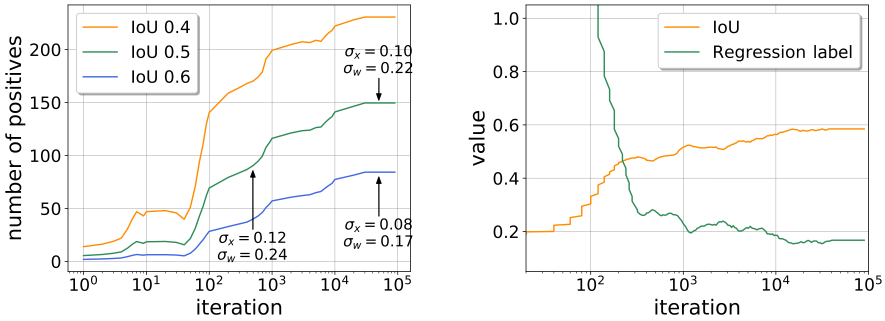
Abstract. Although two-stage object detectors have continuously advanced the state-of-the-art performance in recent years, the training process itself is far from crystal. In this work, we first point out the inconsistency problem between the fixed network settings and the dynamic training procedure, which greatly affects the performance. For example, the fixed label assignment strategy and regression loss function cannot fit the distribution change of proposals and thus are harmful to training high quality detectors. Consequently, we propose *Dynamic R-CNN* to adjust the label assignment criteria (IoU threshold) and the shape of regression loss function (parameters of SmoothL1 Loss) automatically based on the statistics of proposals during training. This dynamic design makes better use of the training samples and pushes the detector to fit more high quality samples. Specifically, our method improves upon ResNet-50-FPN baseline with 1.9% AP and 5.5% AP_{90} on the MS COCO dataset with no extra overhead. Codes and models are available at https://github.com/hkzhang95/DynamicRCNN.

Keywords: dynamic training, high quality object detection

1 Introduction

Benefiting from the advances in deep convolutional neural networks (CNNs) [21,39,15,13], object detection has made remarkable progress in recent years. Modern detection frameworks can be divided into two major categories of one-stage detectors [36,31,28] and two-stage detectors [11,10,37]. And various improvements have been made in recent studies [42,25,46,47,24,23,30,5,19]. In the training procedure of both kinds of pipelines, a classifier and a regressor are adopted respectively to solve the *recognition* and *localization* tasks. Therefore, an effective training process plays a crucial role in achieving high quality object detection[1].

[1] Specifically, high quality represents the results under high IoU.



(a) (b)

Fig. 1. (a) The number of positive proposals under different IoU thresholds during the training process. The curve shows the numbers of positives vary significantly during training, with corresponding changes in regression labels distribution (σ_x and σ_w stands for the standard deviation for x and w respectively). (b) The IoU and regression label of the 75th and 10th most accurate proposals respectively in the training procedure. These curves further show the improved quality of proposals.

Different from the image classification task, the annotations for the classification task in object detection are the ground-truth boxes in the image. So it is not clear how to assign positive and negative labels for the proposals in classifier training since their separation may be ambiguous. The most widely used strategy is to set a threshold for the IoU of the proposal and corresponding ground-truth. As mentioned in Cascade R-CNN [3], training with a certain IoU threshold will lead to a classifier that degrades the performance at other IoUs. However, we cannot directly set a high IoU from the beginning of the training due to the scarcity of positive samples. The solution that Cascade R-CNN provides is to gradually refine the proposals by several stages, which are effective yet time-consuming. As for regressor, the problem is similar. During training, the quality of proposals is improved, however the parameter in SmoothL1 Loss is fixed. Thus it leads to insufficient training for the high quality proposals.

To solve this issue, we first examine an overlooked fact that the quality of proposals is indeed improved during training as shown in Figure 1. We can find that even under different IoU thresholds, the number of positives still increases significantly. Inspired by the illuminating observations, we propose *Dynamic R-CNN*, a simple yet effective method to better exploit the dynamic quality of proposals for object detection. It consists of two components: *Dynamic Label Assignment* and *Dynamic SmoothL1 Loss*, which are designed for classification and regression branches, respectively. First, to train a better classifier that is discriminative for high IoU proposals, we gradually adjust the IoU threshold for positive/negative samples based on the proposals distribution in the training procedure. Specifically, we set the threshold as the IoU of the proposal at a certain percentage since it can reflect the quality of the overall distribution. For regression, we choose to change the shape of the regression loss function to adaptively fit the distribution change of regression label and ensure the contribution of high quality samples to training. In particular, we adjust the β in SmoothL1 Loss based on the regression label distribution, since β actually controls the magnitude of the gradient of small errors (shown in Figure 4).

By this dynamic scheme, we can not only alleviate the data scarcity issue at the beginning of the training, but also harvest the benefit of high IoU training. These two modules explore different parts of the detector, thus could work collaboratively towards high quality object detection. Furthermore, despite the simplicity of our proposed method, Dynamic R-CNN could bring consistent performance gains on MS COCO [29] with almost no extra computational complexity in training. *And during the inference phase, our method does not introduce any additional overhead.* Moreover, extensive experiments verify the proposed method could generalize to other baselines with stronger performance.

2 Related Work

Region-based object detectors. The general practice of region-based object detectors is converting the object detection task into a bounding box classification and a regression problem. In recent years, region-based approaches have been the leading paradigm with top performance. For example, R-CNN [11], Fast R-CNN [10] and Faster R-CNN [37] first generate some candidate region proposals, then randomly sample a small batch with certain foreground-background ratio from all the proposals. These proposals will be fed into a second stage to classify the categories and refine the locations at the same time. Later, some works extended Faster R-CNN to address different problems. R-FCN [7] makes the whole network fully convolutional to improve the speed; and FPN [27] proposes a top-down pathway to combine multi-scale features. Besides, various improvements have been witnessed in recent studies [17,43,25,26,51].

Classification in object detection. Recent researches focus on improving object classifier from various perspectives [28,18,33,41,24,48,6,16]. The classification scores in detection not only determine the semantic category for each proposal, but also imply the localization accuracy, since Non-Maximum Suppression (NMS) suppresses less confident boxes using more reliable ones. It ranks the resultant boxes first using the classification scores. However, as mentioned in IoU-Net [18], the classification score has low correlation with localization accuracy, which leads to noisy ranking and limited performance. Therefore, IoU-Net [18] adopts an extra branch for predicting IoU scores and refining the classification confidence. Softer NMS [16] devises an KL loss to model the variance of bounding box regression directly, and uses that for voting in NMS. Another direction to improve is to raise the IoU threshold for training high quality classifiers, since training with different IoU thresholds will lead to classifiers with corresponding quality. However, as mentioned in Cascade R-CNN [3], directly raising the IoU threshold is impractical due to the vanishing positive samples. Therefore, to produce high quality training samples, some approaches [3,47] adopt sequential stages which are effective yet time-consuming. Essentially, it should be noted that these methods ignore the inherent dynamic property in training procedure which is useful for training high quality classifiers.

Bounding box regression. It has been proved that the performance of models is dependent on the relative weight between losses in multi-task learning [20]. Cascade R-CNN [3] also adopt different regression normalization factors to adjust the aptitude of regression term in different stages. Besides, Libra R-CNN [33] proposes to promote the regression gradients from the accurate samples; and SABL [44] localizes each side of the bounding box with a lightweight two step bucketing scheme for precise localization. However, they mainly focus on a fixed scheme ignoring the dynamic distribution of learning targets during training.

Dynamic training. There are various researches following the idea of dynamic training. A widely used example is adjusting the learning rate based on the training iterations [32]. Besides, Curriculum Learning [1] and Self-paced Learning [22] focus on improving the training order of the examples. Moreover, for object detection, hard mining methods [38,28,33] can also be regarded as a dynamic way. However, they don't handle the core issues in object detection such as constant label assignment strategy. Our method is complementary to theirs.

3 Dynamic Quality in the Training Procedure

Generally speaking, Object detection is complex since it needs to solve two main tasks: *recognition* and *localization*. *Recognition* task needs to distinguish foreground objects from backgrounds and determine the semantic category for them. Besides, the *localization* task needs to find accurate bounding boxes for different objects. To achieve high-quality object detection, we need to further explore the training process of both two tasks as follows.

3.1 Proposal Classification

How to assign labels is an interesting question for the classifier in object detection. It is unique to other classification problems since the annotations are the ground-truth boxes in the image. Obviously, a proposal should be negative if it does not overlap with any ground-truth, and a proposal should be positive if its overlap with a ground-truth is 100%. However, it is a dilemma to define whether a proposal with IoU 0.5 should be labeled as positive or negative.

In Faster R-CNN [37], labels are assigned by comparing the box's highest IoU with ground-truths using a pre-defined IoU threshold. Formally, the paradigm can be formulated as follows (we take a binary classification loss for simplicity):

$$\text{label} = \begin{cases} 1, & \text{if } \max IoU(b, G) \geq T_+ \\ 0, & \text{if } \max IoU(b, G) < T_- \\ -1, & \text{otherwise.} \end{cases} \tag{1}$$

Here b stands for a bounding box, G represents for the set of ground-truths, T_+ and T_- are the positive and negative threshold for IoU. $1, 0, -1$ stand for

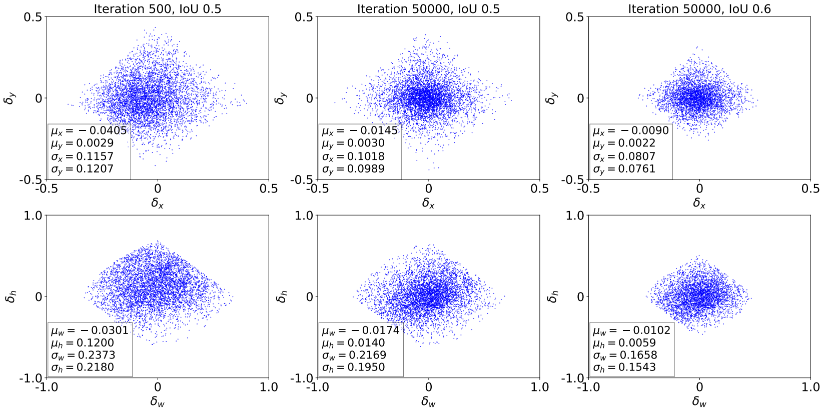


Fig. 2. Δ distribution at different iterations and IoU thresholds (we randomly select some points for simplicity). Column 1&2: under the same IoU threshold, the regression labels are more concentrated as the training goes. Column 2&3: at the same iteration, raising the IoU threshold will significantly change the distribution.

positives, negatives and ignored samples, respectively. As for the second stage of Faster R-CNN, T_+ and T_- are set to 0.5 by default [12]. So the definition of positives and negatives is essentially hand-crafted.

Since the goal of classifier is to distinguish the positives and negatives, training with different IoU thresholds will lead to classifiers with corresponding quality [3].Therefore, to achieve high quality object detection, we need to train the classifier with a high IoU threshold. However, as mentioned in Cascade R-CNN, directly raising the IoU threshold is impractical due to the vanishing positive samples. Cascade R-CNN uses several sequential stages to lift the IoU of the proposals, which are effective yet time-consuming.

So is there a way to get the best of two worlds? As mentioned above, the quality of proposals actually improves along the training. This observation inspires us to take a progressive approach in training: At the beginning, the proposal network is not capable to produce enough high quality proposals, so we use a lower IoU threshold to better accommodate these imperfect proposals in second stage training. As training goes, the quality of proposals improves, we gradually have enough high quality proposals. As a result, we may increase the threshold to better utilize them to train a high quality detector that is more discriminative at higher IoU. We will formulate this process in the following section.

3.2 Bounding Box Regression

The task of bounding box regression is to regress the positive candidate bounding box b to a target ground-truth g. This is learned under the supervision of the regression loss function L_{reg}. To encourage the regression label invariant to scale and location, L_{reg} operates on the offset $\Delta = (\delta_x, \delta_y, \delta_w, \delta_h)$ defined by

$$\begin{aligned} \delta_x &= (g_x - b_x)/b_w, \quad \delta_y = (g_y - b_y)/b_h \\ \delta_w &= \log(g_w/b_w), \quad \delta_h = \log(g_h/b_h). \end{aligned} \tag{2}$$

Since the bounding box regression performs on the offsets, the absolute values of Equation (2) can be very small. To balance the different terms in multi-task learning, $\varDelta$ is usually normalized by pre-defined *mean* and *stdev* (standard deviation) as widely used in many work [37,27,14].

However, we discover that the distribution of regression labels are shifting during training. As shown in Figure 2, we calculate the statistics of the regression labels under different iterations and IoU thresholds. First, from the first two columns, we find that under the same IoU threshold for positives, the *mean* and *stdev* are decreasing as the training goes due to the improved quality of proposals. With the same normalization factors, the contributions of those high quality samples will be reduced based on the definition of SmoothL1 Loss function, which is harmful to the training of high quality regressors. Moreover, with a higher IoU threshold, the quality of positive samples is further enhanced, thus their contributions are reduced even more, which will greatly limit the overall performance. Therefore, to achieve high quality object detection, we need to fit the distribution change and adjust the shape of regression loss function to compensate for the increasing of high quality proposals.

4 Dynamic R-CNN

To better exploit the dynamic property of the training procedure, we propose Dynamic R-CNN which is shown in Figure 3. Our key insight is **adjusting the second stage classifier and regressor to fit the distribution change of proposals**. The two components designed for the classification and localization branch will be elaborated in the following sections.

4.1 Dynamic Label Assignment

The Dynamic Label Assignment (DLA) process is illustrated in Figure 3 (a). Based on the common practice of label assignment in Equation (1) in object detection, the DLA module can be formulated as follows:

$$\text{label} = \begin{cases} 1, & \text{if } \max IoU(b, G) \geq T_{now} \\ 0, & \text{if } \max IoU(b, G) < T_{now}, \end{cases} \tag{3}$$

where T_{now} stands for the current IoU threshold. Considering the dynamic property in training, the distribution of proposals is changing over time. Our DLA updates the T_{now} automatically based on the statistics of proposals to fit this distribution change. Specifically, we first calculate the IoUs I between proposals and their target ground-truths, and then select the K_I-th largest value from I as the threshold T_{now}. As the training goes, T_{now} will increase gradually which

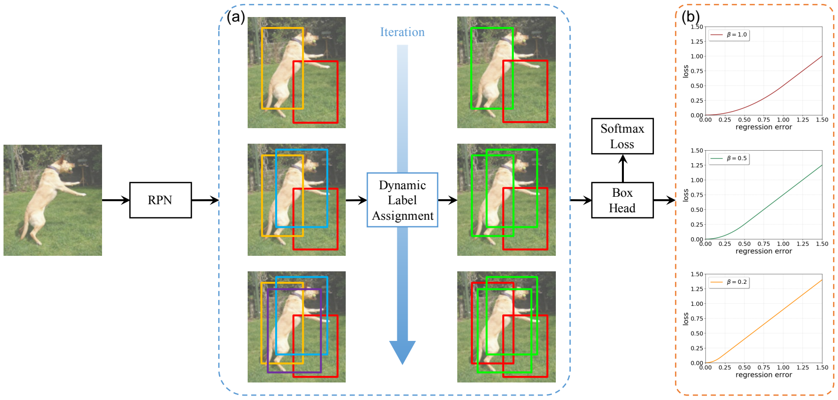


Fig. 3. The overall pipeline of the proposed Dynamic R-CNN. Considering the dynamic property of the training process, Dynamic R-CNN consists of two main components (a) Dynamic Label Assignment (DLA) process and (b) Dynamic SmoothL1 Loss (DSL) from different perspectives. From the left part of (a) we can find that there are more high quality proposals as the training goes. With the improved quality of proposals, DLA will automatically raise the IoU threshold based on the proposal distribution. Then positive (green) and negative (red) labels are assigned for the proposals by DLA which are shown in the right part of (a). Meanwhile, to fit the distribution change and compensate for the increasing of high quality proposals, the shape of regression loss function is also adjusted correspondingly in (b). Best viewed in color.

reflects the improved quality of proposals. In practice, we first calculate the K_I-th largest value in each batch, and then update T_{now} every C iterations using the mean of them to enhance the robustness of the training. It should be noted that the calculation of IoUs is already done by the original method, so there is almost no additional complexity in our method. The resultant IoU thresholds used in training are illustrated in Figure 3 (a).

4.2 Dynamic SmoothL1 Loss

The localization task for object detection is supervised by the commonly used SmoothL1 Loss, which can be formulated as follows:

$$SmoothL1(x, \beta) = \begin{cases} 0.5|x|^2/\beta, & \text{if } |x| < \beta, \\ |x| - 0.5\beta, & \text{otherwise.} \end{cases} \tag{4}$$

Here the x stands for the regression label. β is a hyper-parameter controlling in which range we should use a softer loss function like l_1 loss instead of the original l_2 loss. Considering the robustness of training, β is set default as 1.0 to prevent the exploding loss due to the poor trained network in the early stages. We also illustrate the impact of β in Figure 4, in which changing β leads to different curves of loss and gradient. It is easy to find that a smaller β actually accelerate

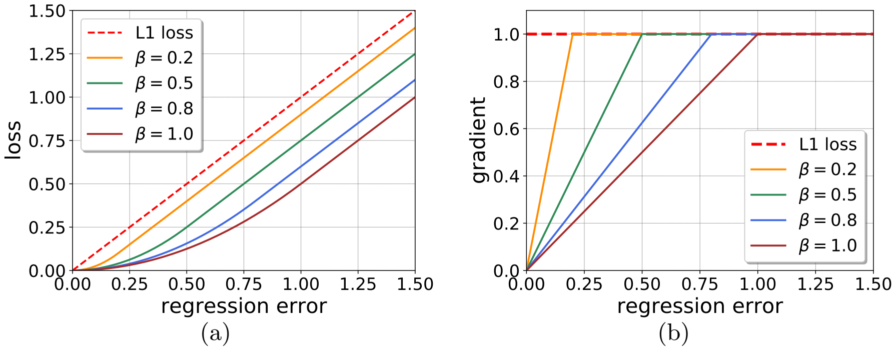


Fig. 4. We show curves for (a) loss and (b) gradient of SmoothL1 Loss with different β here. β is set default as 1.0 in the R-CNN part.

the saturation of the magnitude of gradient, thus it makes more accurate sample contributes more to the network training.

As analyzed in Section 3.2, we need to fit the distribution change and adjust the regression loss function to compensate for the high quality samples. So we propose Dynamic SmoothL1 Loss (DSL) to change the shape of loss function to gradually focus on high quality samples as follows:

$$DSL(x, \beta_{now}) = \begin{cases} 0.5|x|^2/\beta_{now}, & \text{if } |x| < \beta_{now}, \\ |x| - 0.5\beta_{now}, & \text{otherwise.} \end{cases} \tag{5}$$

Similar to DLA, DSL will change the value of β_{now} according to the statistics of regression labels which can reflect the localization accuracy. To be more specific, we first obtain the regression labels E between proposals and their target ground-truths, then select the K_β-th smallest value from E to update the β_{now} in Equation (4). Similarly, we also update the β_{now} every C iterations using the median of the K_β-th smallest label in each batch. We choose median instead of mean as in the classification because we find more outliers in regression labels. Through this dynamic way, appropriate β_{now} will be adopted automatically as shown in Figure 3 (b), which will better exploit the training samples and lead to a high quality regressor.

To summarize the whole method, we describe the proposed Dynamic R-CNN in Algorithm 1. Besides the proposals P and ground-truths G, Dynamic R-CNN has three hyperparamters: IoU threshold top-k K_I, β top-k K_β and update iteration count C. Note that compared with baseline, we only introduce one additional hyperparameter. And we will show soon the results are actually quite robust to the choice of these hyperparameters.

Algorithm 1 Dynamic R-CNN

```
Input:
    Proposal set $P$, ground-truth set $G$.
    IoU threshold top-k $K_I$, $\beta$ top-k $K_\beta$, update iteration count $C$.
Output:
    Trained object detector $D$.
Initialize IoU threshold and SmoothL1 $\beta$ as $T_{now}, \beta_{now}$
Build two empty sets $\mathcal{S}_I, \mathcal{S}_E$ for recording the IoUs and regression labels
for $i = 0$ to max_iter do
    Obtain matched IoUs $I$ and regression labels $E$ between $P$ and $G$
    Select thresholds $I_k, E_k$ based on the $K_I, K_\beta$
    Record corresponding values, add $I_k$ to $\mathcal{S}_I$ and $E_k$ to $\mathcal{S}_E$
    if $i \ \% \ C == 0$ then
        Update IoU threshold: $T_{now} = \text{Mean}(\mathcal{S}_I)$
        Update SmoothL1 $\beta$: $\beta_{now} = \text{Median}(\mathcal{S}_E)$
        $\mathcal{S}_I = \emptyset, \mathcal{S}_E = \emptyset$
    Train the network with $T_{now}, \beta_{now}$
return Improved object detector $D$
```

5 Experiments

5.1 Dataset and Evaluation Metrics

Experimental results are mainly evaluated on the bounding box detection track of the challenging MS COCO [29] 2017 dataset. Following the common practice [28,14], we use the COCO `train` split (∼118k images) for training and report the ablation studies on the `val` split (5k images). We also submit our main results to the evaluation server for the final performance on the `test-dev` split, *which has no disclosed labels*. The COCO-style Average Precision (AP) is chosen as the main evaluation metric which averages AP across IoU thresholds from 0.5 to 0.95 with an interval of 0.05. We also include other metrics to better understand the behavior of the proposed method.

5.2 Implementation Details

For fair comparisons, all experiments are implemented on PyTorch [34] and follow the settings in maskrcnn-benchmark[2] and SimpleDet [4]. We adopt FPN-based Faster R-CNN [37,27] with ResNet-50 [15] model pre-trained on ImageNet [9] as our baseline. All models are trained on the COCO 2017 `train` set and tested on `val` set with image short size at 800 pixels unless noted. Due to the scarcity of positives in the training procedure, we set the NMS threshold of RPN to 0.85 instead of 0.7 for all the experiments.

[2] https://github.com/facebookresearch/maskrcnn-benchmark

Table 1. Comparisons with different baselines (our re-implementations) on COCO `test-dev` set. "MST" and "*" stand for multi-scale training and testing respectively. "2×" and "3×" are training schedules which extend the iterations by 2/3 times.

Method	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster R-CNN	ResNet-50	37.3	58.5	40.6	20.3	39.2	49.1
Faster R-CNN+2×	ResNet-50	38.1	58.9	41.5	20.5	40.0	50.0
Faster R-CNN	ResNet-101	39.3	60.5	42.7	21.3	41.8	51.7
Faster R-CNN+2×	ResNet-101	39.9	60.6	43.5	21.4	42.4	52.1
Faster R-CNN+3×+MST	ResNet-101	42.8	63.8	46.8	24.8	45.6	55.6
Faster R-CNN+3×+MST	ResNet-101-DCN	44.8	65.5	48.8	26.2	47.6	58.1
Faster R-CNN+3×+MST*	ResNet-101-DCN	46.9	68.1	51.4	30.6	49.6	58.1
Dynamic R-CNN	ResNet-50	39.1	58.0	42.8	21.3	40.9	50.3
Dynamic R-CNN+2×	ResNet-50	39.9	58.6	43.7	21.6	41.5	51.9
Dynamic R-CNN	ResNet-101	41.2	60.1	45.1	22.5	43.6	53.2
Dynamic R-CNN+2×	ResNet-101	42.0	60.7	45.9	22.7	44.3	54.3
Dynamic R-CNN+3×+MST	ResNet-101	44.7	63.6	49.1	26.0	47.4	57.2
Dynamic R-CNN+3×+MST	ResNet-101-DCN	46.9	65.9	51.3	28.1	49.6	60.0
Dynamic R-CNN+3×+MST*	ResNet-101-DCN	49.2	68.6	54.0	32.5	51.7	60.3

5.3 Main Results

We compare Dynamic R-CNN with corresponding baselines on COCO `test-dev` set in Table 1. For fair comparisons, We report our re-implemented results.

First, we prove that our method can work on different backbones. Dynamic R-CNN achieves 39.1% AP with ResNet-50 [15], which is 1.8 points higher than the FPN-based Faster R-CNN baseline. With a stronger backbone like ResNet-101, Dynamic R-CNN can also achieve consistent gains (+1.9 points).

Then, our dynamic design is also compatible with other training and testing skills. The results are consistently improved by progressively adding in 2× longer training schedule, multi-scale training (extra 1.5× longer training schedule), multi-scale testing and deformable convolution [51]. With the best combination, out Dynamic R-CNN achieves 49.2% AP, which is still 2.3 points higher than the Faster R-CNN baseline.

These results show the effectiveness and robustness of our method since it can work together with different backbones and multiple training and testing skills. It should also be noted that the performance gains are almost free.

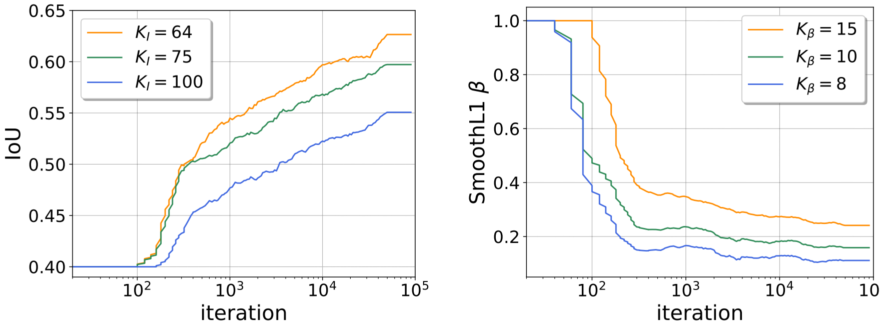
5.4 Ablation Experiments

To show the effectiveness of each proposed component, we report the overall ablation studies in Table 2.

1) Dynamic Label Assignment (DLA). DLA brings 1.2 points higher box AP than the ResNet-50-FPN baseline. To be more specific, results in higher IoU

Table 2. Results of each component in Dynamic R-CNN on COCO `val` set.

Backbone	DLA	DSL	AP	ΔAP	AP_{50}	AP_{60}	AP_{70}	AP_{80}	AP_{90}
ResNet-50-FPN			37.0	-	58.0	53.5	46.0	32.6	9.7
ResNet-50-FPN		✓	38.0	+1.0	57.6	53.5	46.7	34.4	13.2
ResNet-50-FPN	✓		38.2	+1.2	57.5	53.6	47.1	35.2	12.6
ResNet-50-FPN	✓	✓	38.9	+1.9	57.3	53.6	47.4	36.3	15.2



(a)

(b)

Fig. 5. Trends of (a) IoU threshold and (b) SmoothL1 β under different settings based on our method. Obviously the distribution has changed a lot during training.

metrics are consistently improved, especially for the 2.9 points gains in AP_{90}. It proves the effectiveness of our method for pushing the classifier to be more discriminative at higher IoU thresholds.

2) Dynamic SmoothL1 Loss (DSL). DSL improves the box AP from 37.0 to 38.0. Results in higher IoU metrics like AP_{80} and AP_{90} are hugely improved, which validates the effectiveness of changing the loss function to compensate for the high quality samples during training. Moreover, as analyzed in Section 3.2, with DLA the quality of positives is further improved thus their contributions are reduced even more. So applying DSL on DLA will also bring reasonable gains especially on high quality metrics. **To sum up, Dynamic R-CNN improves the baseline by 1.9 points** AP **and 5.5 points** AP_{90}.

3) Illustration of dynamic training. To further illustrate the dynamics in the training procedure, we show the trends of IoU threshold and SmoothL1 β under different settings based on our method in Figure 5. Here we clip the values of IoU threshold and β to 0.4 and 1.0 respectively at the beginning of training. Regardless of the specific values of K_I and K_β, the overall trend of IoU threshold is increasing while that for SmoothL1 β is decreasing during training. These results again verify the proposed method work as expected.

Table 3. Ablation study on K_I.

K_I	AP	AP$_{50}$	AP$_{60}$	AP$_{70}$	AP$_{80}$	AP$_{90}$
-	37.0	58.0	53.5	46.0	32.6	9.7
64	38.1	57.2	53.3	46.8	35.1	**12.8**
75	**38.2**	57.5	53.6	**47.1**	**35.2**	12.6
100	37.9	**57.9**	**53.8**	46.9	34.2	11.6

Table 4. Ablation study on C.

C	AP	AP$_{50}$	AP$_{60}$	AP$_{70}$	AP$_{80}$	AP$_{90}$
-	37.0	58.0	53.5	46.0	32.6	9.7
20	38.0	57.4	53.5	47.0	35.0	12.5
100	38.2	57.5	53.6	47.1	35.2	12.6
500	38.1	57.6	53.5	47.2	34.8	12.6

Table 5. Ablation study on K_β.

Setting	AP	AP$_{50}$	AP$_{60}$	AP$_{70}$	AP$_{80}$	AP$_{90}$
$\beta = 1.0$	37.0	58.0	53.5	46.0	32.6	9.7
$\beta = 2.0$	35.9	57.7	53.2	45.1	30.1	8.3
$\beta = 0.5$	37.5	57.6	53.3	46.4	33.5	11.3
$K_\beta = 15$	37.6	57.3	53.1	46.0	33.9	12.5
$K_\beta = 10$	38.0	57.6	53.5	46.7	34.4	13.2
$K_\beta = 8$	37.6	57.5	53.3	45.9	33.9	12.4

Table 6. Inference speed comparisons using ResNet-50-FPN backbone on RTX 2080TI GPU.

Method	FPS
Dynamic R-CNN	13.9
Cascade R-CNN	11.2
Dynamic Mask R-CNN	11.3
Cascade Mask R-CNN	7.3

5.5 Studies on the effect of hyperparameters

Ablation study on K_I in DLA. Experimental results on different K_I are shown in Table 3. Compared to the baseline, DLA can achieve consistent gains in AP regardless of the choice of K_I. These results prove the universality of K_I. Moreover, the performance on various metrics are changed under different K_I. Choosing K_I as 64/75/100 means that nearly 12.5%/15%/20% of the whole batch are selected as positives. Generally speaking, setting a smaller K_I will increase the quality of selected samples, which will lead to better accuracy under higher metrics like AP$_{90}$. On the contrary, adopting a larger K_I will be more helpful for the metrics at lower IoU. Finally, we find that setting K_I as 75 achieves the best trade-off and use it as the default value for further experiments. All these ablations prove the effectiveness and robustness of the DLA part.

Ablation study on K_β in DSL. As shown in Table 5, we first try different β on Faster R-CNN and empirically find that a smaller β leads to better performance. Then, experiments under different K_β are provided to show the effects of K_β. Regardless of the certain value of K_β, DSL can achieve consistent improvements compared with various fine-tuned baselines. Specifically, with our best setting, DSL can bring 1.0 point higher AP than the baseline, and the improvement mainly lies in the high quality metrics like AP$_{90}$ (+3.5 points). These experimental results prove that our DSL is effective in compensating for high quality samples and can lead to a better regressor due to the advanced dynamic design.

Ablation study on iteration count C. Due to the concern of robustness, we update T_{now} and β_{now} every C iterations using the statistics in the last interval. To show the effects of different iteration count C, we try different values of C on the proposed method. As shown in Table 4, setting C as 20, 100 and 500 leads to very similar results, which proves the robustness to this hyperparameter.

Table 7. The universality of Dynamic R-CNN. We apply the idea of dynamic training on Mask R-CNN under different backbones. “bbox” and “segm” stand for object detection and instance segmentation results on COCO `val` set, respectively.

Backbone	+Dynamic	AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{segm}	AP^{segm}_{50}	AP^{segm}_{75}
ResNet-50-FPN		37.5	58.0	40.7	33.8	54.6	36.0
	✓	39.4	57.6	43.3	34.8	55.0	37.5
ResNet-101-FPN		39.7	60.7	43.2	35.6	56.9	37.7
	✓	41.8	60.4	45.8	36.7	57.5	39.4

Complexity and speed. As shown in Algorithm 1, the main computational complexity of our method lies in the calculations of IoUs and regression labels, which are already done by the original method. Thus the additional overhead only lies in calculating the mean or median of a short vector, which basically **does not increase the training time**. Moreover, since our method only changes the training procedure, obviously the inference speed will not be slowed down.

Our advantage compared to other high quality detectors like Cascade R-CNN is the efficiency. Cascade R-CNN increases the training time and slows down the inference speed while our method does not. Specifically, as shown in Table 6, Dynamic R-CNN achieves 13.9 FPS, which is ∼1.25 times faster than Cascade R-CNN (11.2 FPS) under ResNet-50-FPN backbone. Moreover, with larger heads, the cascade manner will further slow down the speed. Dynamic Mask R-CNN runs ∼1.5 times faster than Cascade Mask R-CNN. Note that the difference will be more apparent as the backbone gets smaller (∼1.74 times faster, 13.6 FPS vs 7.8 FPS under ResNet-18 backbone with mask head), since the main overhead of Cascade R-CNN is the two additional headers.

5.6 Universality

Since the viewpoint of dynamic training is a general concept, we believe that it can be adopted in different methods. To validate the universality, we further apply the dynamic design on Mask R-CNN with different backbones. As shown in Table 7, adopting the dynamic design can not only bring ∼2.0 points higher box AP but also improve the instance segmentation results regardless of backbones. Note that we only adopt the DLA and DSL which are designed for object detection, so these results further demonstrate the universality and effectiveness of our dynamic design on improving training procedure for current detectors.

5.7 Comparison with State-of-the-Arts

We compare Dynamic R-CNN with the state-of-the-art object detectors on COCO `test-dev` set in Table 8. Considering that various backbones and training/testing settings are adopted by different detectors (including deformable

Table 8. Comparisons of single-model results on COCO `test-dev` set.

Method	Backbone	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
RetinaNet [28]	ResNet-101	39.1	59.1	42.3	21.8	42.7	50.2
CornerNet [23]	Hourglass-104	40.5	56.5	43.1	19.4	42.7	53.9
FCOS [42]	ResNet-101	41.0	60.7	44.1	24.0	44.1	51.0
FreeAnchor [49]	ResNet-101	41.8	61.1	44.9	22.6	44.7	53.9
RepPoints [46]	ResNet-101-DCN	45.0	66.1	49.0	26.6	48.6	57.5
CenterNet [50]	Hourglass-104	45.1	63.9	49.3	26.6	47.1	57.7
ATSS [48]	ResNet-101-DCN	46.3	64.7	50.4	27.7	49.8	58.4
Faster R-CNN [27]	ResNet-101	36.2	59.1	39.0	18.2	39.0	48.2
Mask R-CNN [14]	ResNet-101	38.2	60.3	41.7	20.1	41.1	50.2
Regionlets [45]	ResNet-101	39.3	59.8	-	21.7	43.7	50.9
Libra R-CNN [33]	ResNet-101	41.1	62.1	44.7	23.4	43.7	52.5
Cascade R-CNN [3]	ResNet-101	42.8	62.1	46.3	23.7	45.5	55.2
SNIP [40]	ResNet-101-DCN	44.4	66.2	49.9	27.3	47.4	56.9
DCNv2 [51]	ResNet-101-DCN	46.0	67.9	50.8	27.8	49.1	59.5
TridentNet [25]	ResNet-101-DCN	48.4	69.7	53.5	31.8	51.3	60.3
Dynamic R-CNN	ResNet-101	42.0	60.7	45.9	22.7	44.3	54.3
Dynamic R-CNN*	ResNet-101-DCN	50.1	68.3	55.6	32.8	53.0	61.2

convolutions [8,51], image pyramid scheme [40], large-batch Batch Normalization [35] and Soft-NMS [2]), we report the results of our method with two types.

Dynamic R-CNN applies our method on FPN-based Faster R-CNN with ResNet-101 as backbone, and it can achieve 42.0% AP without bells and whistles. Dynamic R-CNN* adopts image pyramid scheme (multi-scale training and testing), deformable convolutions and Soft-NMS. It further improves the results to 50.1% AP, outperforming all the previous detectors.

6 Conclusion

In this paper, we take a thorough analysis of the training process of detectors and find that the fixed scheme limits the overall performance. Based on the advanced dynamic viewpoint, we propose Dynamic R-CNN to better exploit the training procedure. With the help of the simple but effective components like Dynamic Label Assignment and Dynamic SmoothL1 Loss, Dynamic R-CNN brings significant improvements on the challenging COCO dataset with no extra cost. Extensive experiments with various detectors and backbones validate the universality and effectiveness of Dynamic R-CNN. We hope that this dynamic viewpoint can inspire further researches in the future.

Acknowledgements This work is partially supported by Natural Science Foundation of China (NSFC): 61876171 and 61976203, and Beijing Natural Science Foundation under Grant L182054.

References

1. Bengio, Y., Louradour, J., Collobert, R., Weston, J.: Curriculum learning. In: ICML (2009)
2. Bodla, N., Singh, B., Chellappa, R., Davis, L.S.: Soft-NMS – improving object detection with one line of code. In: ICCV (2017)
3. Cai, Z., Vasconcelos, N.: Cascade R-CNN: Delving into high quality object detection. In: CVPR (2018)
4. Chen, Y., Han, C., Li, Y., Huang, Z., Jiang, Y., Wang, N., Zhang, Z.: SimpleDet: A simple and versatile distributed framework for object detection and instance recognition. JMLR **20**(156), 1–8 (2019)
5. Chen, Y., Han, C., Wang, N., Zhang, Z.: Revisiting feature alignment for one-stage object detection. arXiv:1908.01570 (2019)
6. Cheng, B., Wei, Y., Shi, H., Feris, R., Xiong, J., Huang, T.: Revisiting RCNN: On awakening the classification power of faster RCNN. In: ECCV (2018)
7. Dai, J., Li, Y., He, K., Sun, J.: R-FCN: Object detection via region-based fully convolutional networks. In: NIPS (2016)
8. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: ICCV (2017)
9. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A large-scale hierarchical image database. In: CVPR (2009)
10. Girshick, R.: Fast R-CNN. In: ICCV (2015)
11. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
12. Girshick, R., Radosavovic, I., Gkioxari, G., Dollár, P., He, K.: Detectron. https://github.com/facebookresearch/detectron (2018)
13. Gu, X., Chang, H., Ma, B., Zhang, H., Chen, X.: Appearance-preserving 3d convolution for video-based person re-identification. In: ECCV (2020)
14. He, K., Gkioxari, G., Dollar, P., Girshick, R.: Mask R-CNN. In: ICCV (2017)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
16. He, Y., Zhu, C., Wang, J., Savvides, M., Zhang, X.: Bounding box regression with uncertainty for accurate object detection. In: CVPR (2019)
17. Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., Murphy, K.: Speed/accuracy trade-offs for modern convolutional object detectors. In: CVPR (2017)
18. Jiang, B., Luo, R., Mao, J., Xiao, T., Jiang, Y.: Acquisition of localization confidence for accurate object detection. In: ECCV (2018)
19. Jiang, Z., Liu, Y., Yang, C., Liu, J., Gao, P., Zhang, Q., Xiang, S., Pan, C.: Learning where to focus for efficient video object detection. In: ECCV (2020)
20. Kendall, A., Gal, Y., Cipolla, R.: Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: CVPR (2018)
21. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: NIPS (2012)
22. Kumar, M.P., Packer, B., Koller, D.: Self-paced learning for latent variable models. In: NIPS (2010)
23. Law, H., Deng, J.: CornerNet: Detecting objects as paired keypoints. In: ECCV (2018)
24. Li, H., Wu, Z., Zhu, C., Xiong, C., Socher, R., Davis, L.S.: Learning from noisy anchors for one-stage object detection. In: CVPR (2020)

25. Li, Y., Chen, Y., Wang, N., Zhang, Z.: Scale-aware trident networks for object detection. In: ICCV (2019)
26. Li, Z., Peng, C., Yu, G., Zhang, X., Deng, Y., Sun, J.: DetNet: Design backbone for object detection. In: ECCV (2018)
27. Lin, T.Y., Dollar, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR (2017)
28. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollar, P.: Focal loss for dense object detection. In: ICCV (2017)
29. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: ECCV (2014)
30. Liu, S., Huang, D., Wang, Y.: Receptive field block net for accurate and fast object detection. In: ECCV (2018)
31. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: SSD: Single shot multibox detector. In: ECCV (2016)
32. Loshchilov, I., Hutter, F.: SGDR: Stochastic gradient descent with warm restarts. In: ICLR (2017)
33. Pang, J., Chen, K., Shi, J., Feng, H., Ouyang, W., Lin, D.: Libra R-CNN: Towards balanced learning for object detection. In: CVPR (2019)
34. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in PyTorch. In: NIPS Workshop (2017)
35. Peng, C., Xiao, T., Li, Z., Jiang, Y., Zhang, X., Jia, K., Yu, G., Sun, J.: MegDet: A large mini-batch object detector. In: CVPR (2018)
36. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: CVPR (2016)
37. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS (2015)
38. Shrivastava, A., Gupta, A., Girshick, R.: Training region-based object detectors with online hard example mining. In: CVPR (2016)
39. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
40. Singh, B., Davis, L.S.: An analysis of scale invariance in object detection - SNIP. In: CVPR (2018)
41. Tan, Z., Nie, X., Qian, Q., Li, N., Li, H.: Learning to rank proposals for object detection. In: ICCV (2019)
42. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: Fully convolutional one-stage object detection. In: ICCV (2019)
43. Wang, J., Chen, K., Yang, S., Loy, C.C., Lin, D.: Region proposal by guided anchoring. In: CVPR (2019)
44. Wang, J., Zhang, W., Cao, Y., Chen, K., Pang, J., Gong, T., Shi, J., Loy, C.C., Lin, D.: Side-aware boundary localization for more precise object detection. In: ECCV (2020)
45. Xu, H., Lv, X., Wang, X., Ren, Z., Bodla, N., Chellappa, R.: Deep regionlets for object detection. In: ECCV (2018)
46. Yang, Z., Liu, S., Hu, H., Wang, L., Lin, S.: RepPoints: Point set representation for object detection. In: ICCV (2019)
47. Zhang, H., Chang, H., Ma, B., Shan, S., Chen, X.: Cascade RetinaNet: Maintaining consistency for single-stage object detection. In: BMVC (2019)
48. Zhang, S., Chi, C., Yao, Y., Lei, Z., Li, S.Z.: Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In: CVPR (2020)

49. Zhang, X., Wan, F., Liu, C., Ji, R., Ye, Q.: FreeAnchor: Learning to match anchors for visual object detection. In: NeurIPS (2019)
50. Zhou, X., Wang, D., Krähenbühl, P.: Objects as points. arXiv:1904.07850 (2019)
51. Zhu, X., Hu, H., Lin, S., Dai, J.: Deformable convnets v2: More deformable, better results. In: CVPR (2019)