Dynamic Sparse R-CNN

Qinghang Hong*, Fengming Liu*, Dong Li, Ji Liu, Lu Tian, Yi Shan
Advanced Micro Devices, Inc., Beijing, China
{d.li, lu.tian, yi.shan}@amd.com

Abstract

Sparse R-CNN is a recent strong object detection baseline by set prediction on sparse, learnable proposal boxes and proposal features. In this work, we propose to improve Sparse R-CNN with two dynamic designs. First, Sparse R-CNN adopts a one-to-one label assignment scheme, where the Hungarian algorithm is applied to match only one positive sample for each ground truth. Such one-to-one assignment may not be optimal for the matching between the learned proposal boxes and ground truths. To address this problem, we propose dynamic label assignment (DLA) based on the optimal transport algorithm to assign increasing positive samples in the iterative training stages of Sparse R-CNN. We constrain the matching to be gradually looser in the sequential stages as the later stage produces the refined proposals with improved precision. Second, the learned proposal boxes and features remain fixed for different images in the inference process of Sparse R-CNN. Motivated by dynamic convolution, we propose dynamic proposal generation (DPG) to assemble multiple proposal experts dynamically for providing better initial proposal boxes and features for the consecutive training stages. DPG thereby can derive sample-dependent proposal boxes and features for inference. Experiments demonstrate that our method, named Dynamic Sparse R-CNN, can boost the strong Sparse R-CNN baseline with different backbones for object detection. Particularly, Dynamic Sparse R-CNN reaches the state-of-the-art 47.2% AP on the COCO 2017 validation set, surpassing Sparse R-CNN by 2.2% AP with the same ResNet-50 backbone.

1. Introduction

Object detection is a fundamental task in computer vision which aims at predicting a set of objects with locations and corresponding pre-defined categories in a given image. It has been widely applied in multiple fields including intelligent surveillance and autonomous driving. Object detection has witnessed a rapid development in the recent years, with varying feature extraction backbones from convolutional neural network (CNN) [7, 11, 24, 27] to Transformer [6, 18] and varying detection pipeline designs [2, 17, 22, 23, 25, 28]. The detectors can mainly be divided into one-stage, two-stage and multi-stage methods according to the regression times. One-stage detectors [17, 28] directly predict the regression targets and categories of objects in a given image without the refinement step. Two-stage detectors [4, 9, 14, 23] first generate a limited number of candidate proposals for foreground (e.g., region proposal network (RPN)) and then pass the proposals to the detection network to refine the location and category. Multi-stage detectors [1] would refine the location and category multiple times with improved performance but often require large computation overhead. One-stage methods generally can be divided into anchor-based and anchor-free detectors. Anchor-based detectors [15, 17, 22] design dense pre-defined anchors, tile anchors across the image, and then directly predict category and refine the coordinates of anchors. However, manual anchor configurations could be sub-optimal for the final performance. Anchor-free detectors [13, 28] are proposed to overcome this issue by removing the anchor design. They

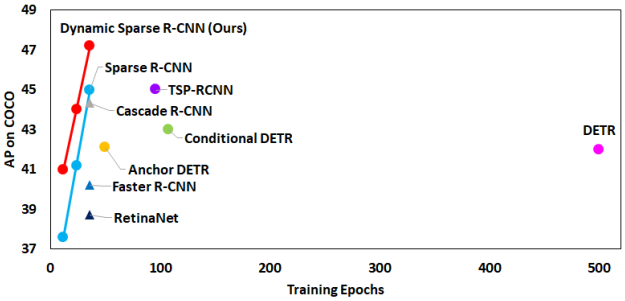


Figure 1. Performance vs. training epochs on the COCO 2017 validation set. All results are reported with single-scale inference using the ResNet-50 backbone. Our Dynamic Sparse R-CNN achieves the state-of-the-art 47.2% AP with the same 36 training epochs as Sparse R-CNN. Circles: Transformer-based methods. Triangles: CNN-based methods.

*Equal contribution.

typically use center points or regions inside ground truth to define positive proposals and predict offsets to obtain final bounding boxes.

Recently, Transformer-based detectors [2, 20, 25, 29] have been proposed by formulating object detection as a set prediction problem using the Transformer encoder and decoder architecture. These methods replace anchor mechanisms with a small number of learnable object queries, which can model the relationships between objects and global image context to output the final predictions. Hungarian algorithm is used to find a bipartite matching between ground truths and predictions based on the combined loss of classification and regression. The label assignment in these detectors is a one-to-one way where only one single detection matches one ground truth during training.

Motivated by existing CNN-based methods using many-to-one label assignment schemes [8, 15, 28], We assume that assigning multiple positives to a GT can optimize the proposals more efficiently and can promote the detector training for better performance. Thus, we propose dynamic label assignment (DLA) with many-to-one matching based on the optimal transport algorithm for the strong baseline of Sparse R-CNN. We also adopt gradually increasing positive samples assigned to GTs in the iterative stages of Sparse R-CNN. Since each stage produces refined proposal boxes and features for the next one, we expect to constrain the matching between GTs and prediction boxes to be stricter in the early stages and looser in the later stages owing to the increasing precision of predictions in the sequential stages. Moreover, in Sparse R-CNN, the object queries (i.e., proposal boxes and proposal features) are learnable during training but remain fixed for different images during inference. Motivated by dynamic convolution [3], we propose dynamic proposal generation (DPG) to provide better initial proposal boxes and features in the first iterative stage. Compared to fixed proposals, DPG can aggregate multiple parallel proposal experts which are sample-dependent and output dynamic proposals for inference. We name our method as Dynamic Sparse R-CNN, which reaches the state-of-the-art 47.2% AP on the COCO 2017 validation set, surpassing the Sparse R-CNN baseline by a large margin of 2.2% AP with the same ResNet-50 backbone (Figure 1).

Our main contributions can be summarized as follows. (1) We point out that many-to-one label assignment in Transformer-based detection is more reasonable and effective than the one-to-one scheme. We apply the optimal transport assignment method into Sparse R-CNN and assign gradually increasing positive samples to GTs in the iterative stages. (2) We design a dynamic proposal generation mechanism to learn multiple proposal experts and assemble them for generating dynamic proposal boxes and features for inference. (3) We integrate the two dynamic designs into Sparse R-CNN and the resulting Dynamic Sparse R-CNN detector obtains a large AP gain of 2.2%, reaching the state-of-the-art 47.2% AP on the COCO validation set with ResNet-50.

2. Related Work

2.1. General Object Detection

CNN-based detectors have achieved great progress owing to the development of various feature extraction backbones and pipeline designs. One-stage detectors directly predict the location and associated categories of object in a given image without region proposal and refinement components, including anchor-based [15, 17, 22] and anchor-free [13, 28] methods. Two-stage detectors [4, 14, 23] first generate a fixed number of proposal for foreground with region proposal network (RPN) and then pass the proposals to the detection network for refining the locations and categories of objects.

Recently, Transformer-based detectors [2, 20, 29, 35] utilizes Transformer encoder and decoder architecture to reformulate the object detection as a set prediction problem. They design a small number of learnable object queries to model the relations between objects and the global image context, and have shown impressive performance. Object queries in decoders are a required component of DETR [2] (7.8% AP drops without them). Conditional DETR [20] proposes a conditional spatial query for fast training convergence. Anchor DETR [29] proposes a query design based on anchor points and achieve near performance to DETR with less training time. Sparse R-CNN [25] proposes learnable proposal boxes and proposal features, and pass the RoI features extracted on the feature map (based on proposal boxes) and associated proposal features to the iterative structure (i.e., dynamic head) for prediction.

2.2. Label Assignment

Label Assignment plays a prominent part in modern object detectors. Anchor-based detectors [15, 17, 23] usually adopt IoU at a certain threshold as the assigning criterion. For example, RetinaNet defines the anchors having IoU score higher than 0.5 as positive samples and others as negative samples. YOLO detectors [21, 22] only adopt the anchor having the max IoU score associated to the ground-truth as the positive sample and such label assignment is a one-to-one matching method. Anchor-Free detectors [13, 28, 34] define center points or shrinking center regions of ground truth as positives and take others as negatives. ATSS [32] indicates that the essential difference between anchor-based and anchor-free detectors is label assignment. It proposes an adaptive training sample selection method which divides positive and negative samples according to statistical characteristics of object. PAA [12] proposes a probabilistic anchor assignment method

by modeling the distribution of joint loss for positive and negative samples as the Gaussian distribution. OTA [8] formulates the label assignment as an optiomal transport problem by defining ground truths and background as supplier and defining anchors as demander, and then employs Sinkhorn-Knopp Iteration to efficiently optimize the problem. Transformer-based detectors [2, 20, 25, 29, 35] formulate object detection as a set prediction problem and treat label assignment between ground truths and object queries as a bipartite matching. Hungarian algorithm is used to optimize the one-to-one matching between ground truths and object queries by minimizing the global loss. In this paper, we assume that one-to-one label assignment is sub-optimal in Transformer-based detectors and explore a dynamic label assignment with many-to-one matching for Sparse R-CNN inspired by OTA [8].

2.3. Dynamic Convolution

Dynamic convolution [3] is a technique that dynamically combines multiple convolution kernels with learnable sample-dependent weights to enhance the representation capability of the model. Temperature annealing in softmax can help improve both the training efficiency and final performance. CondConv [31] proposes conditionally parameterized convolutions, which learn specialized convolution kernels for each input image. It combines multiple convolution kernels with weights generated with sub-net using sigmoid transformation to construct a image-specified convolution kernel. DyNet [33] designs several dynamic convolution neural networks based on dynamic convolution including Dy-mobile, Dy-shuffle and Dy-ResNet, etc. In this work, we analyze that the fixed proposal boxes and features in Sparse R-CNN for different inputs during inference is sub-optimal and inflexible. Motivated by dynamic convolution, we improve Sparse R-CNN by generating dynamic sample-dependent proposals during inference.

3. Proposed Approach

3.1. Revisit Sparse R-CNN

Sparse R-CNN [25] is a recent strong object detection baseline by set prediction on a sparse set of learnable object proposals. It uses an iterative structure (i.e., dynamic head) to gradually produce and refine the predictions. The input of each iterative stage consists of three parts: FPN features extracted by the backbone, proposal boxes and proposal features. The output includes the predicted boxes, the corresponding classes and object features of the boxes. The predicted boxes and object features output by one stage are respectively used as the refined proposal boxes and proposal features to the next stage. Proposal boxes are a small fixed set of region proposals ($N_p \times 4$), indicating the potential locations of the objects. Proposal features are latent vectors ($N_p \times C$) to encode the instance characteristics (e.g., pose and shape). In Sparse R-CNN, proposal boxes are learned during training and fixed for inference. Sparse R-CNN applies the set-based loss to produce a bipartite matching between predictions and ground truth objects, which uses one-to-one matching with the Hungarian algorithm. Figure 2 (a) illustrates the design of Sparse R-CNN.

We analyze two main limitations of Sparse R-CNN as follows. First, Sparse R-CNN adopts one-to-one matching between the detection predictions and the ground truths, which is likely to sub-optimal and inefficient for training. Second, the learned proposal boxes and proposal features in Sparse R-CNN represent the statistics of the training set, which are not adaptive for a specific test image. In our work, we devise two modifications to improve Sparse R-CNN. Figure 2 gives the overview of our method and we introduce algorithm details in the following sections.

3.2. Dynamic Label Assignment

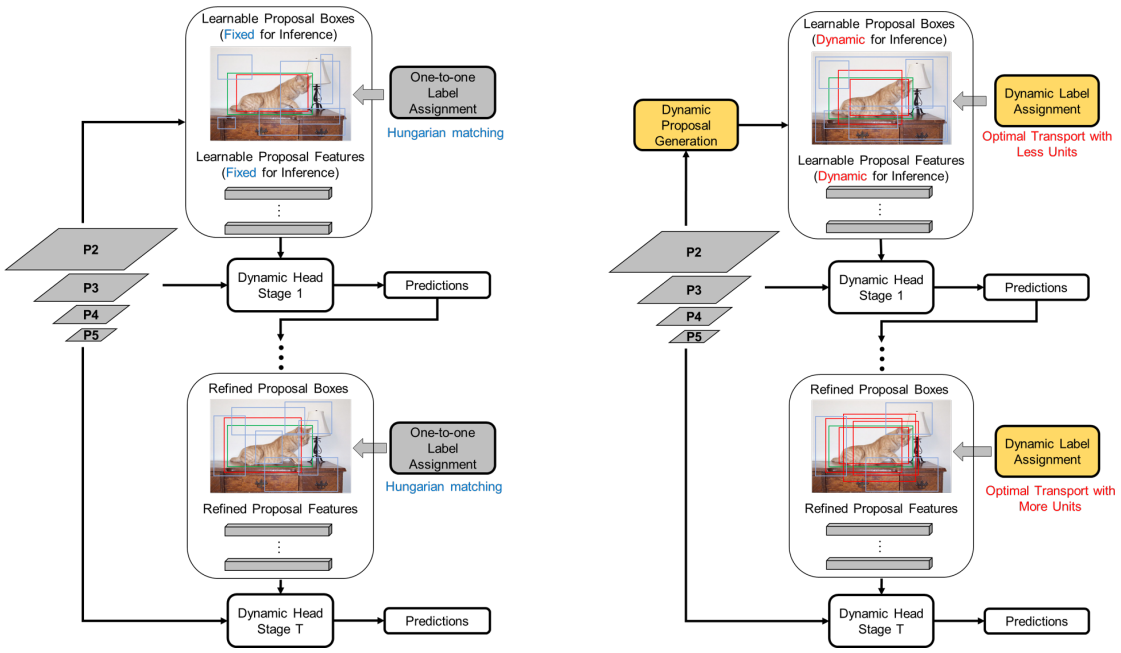
In Sparse R-CNN, the Hungarian algorithm is used for one-to-one matching, where each ground truth is matched to one predicted box. We assume that such one-to-one matching is likely to be sub-optimal. Assigning multiple positives to a GT can optimize the proposals more efficiently and promote the detector training.

To implement many-to-one matching, we follow the CNN-based method [8] and apply the optimal transport assignment (OTA) in Transformer. Specifically, OTA is a formulation that explores how the detection boxes should be matched to ground truths. The formulation treats the ground truths as suppliers to provide quota for assignment, and treats detection boxes as demanders to seek for assignments. The background class is also formulated as a supplier that provides default assignment.

Mathematically, suppose that we have m ground truths in an image and each provides $s_i = k$ assignments, which are referred as *units*. Each of n detection boxes tries to get an unit and a successful matching is referred as a positive assignment. The background provides $s_i = n - k * m$ units to fulfill detection boxes that are not assigned to any ground truth, which is referred as negative assignments. The optimization target can be defined as follows.

$$
\begin{aligned}
&\min_{\pi} \sum_{i=1}^{m} \sum_{j=1}^{n} C(i,j) * \pi(i,j), \\
&\text{s.t.} \sum_{i=1}^{m} \pi(i,j) = 1, \ \sum_{j=1}^{n} \pi(i,j) = s_i, \ \sum_{i=1}^{m} s_i = n, \\
&\pi(i,j) > 0, i = 1,2,...,m, j = 1,2,...,n, \\
&C(i,j) = \begin{cases} L_{cls}(i,j) + \alpha * L_{reg}(i,j), \text{positive assignment} \\ L_{cls}(background, j), \text{negative assignment} \end{cases}
\end{aligned}
\tag{1}
$$

where i is the index of ground truth, j is the index of detection boxes ($j = 1, ...n$), α is a coefficient balancing the



(a) Sparse R-CNN (b) Dynamic Sparse R-CNN

Figure 2. Comparisons with (a) the Sparse R-CNN baseline and (b) our Dynamic Sparse R-CNN. Sparse R-CNN uses one-to-one label assignment optimized by Hungarian algorithm and fixed proposal boxes / features during inference. Dynamic Sparse R-CNN improve Sparse R-CNN with two dynamic designs. First, we adopt dynamic many-to-one label assignment optimized by the optimal transport algorithm with unit increasing strategy. Second, we propose dynamic proposal generation to generate sample-dependent proposal boxes and features.

classification and regression losses. The cost of each positive assignment is the sum of the classification loss L_{cls} and regression loss L_{reg}, while the cost of each negative assignment is only the classification loss. $\pi(i, j)$ represents the matching result to be optimized between ground truth i and detection box j.

The number of units k offered by each supplier can be fixed or dynamic. Following the Dynamic k Estimation method in [8], our work dynamically estimates the k value based on the IoU between the predictions and the ground-truth boxes. In this strategy, top q IoU values for each ground truth are selected and summed up (and converted to an integer) as the estimation for the k value. Based on the optimal transport theory for label assignment ($\sum_{i=1}^{m} \pi(i, j) = 1$ in Eq. 1), each proposal (i.e., demander) only needs one unit of label provided by GT (i.e., supplier). Thus, one proposal will not be assigned to different GTs. The Dynamic k Estimation method generally holds $k < q$. Suppose that m is the number of GTs and N_p is the number of total proposals, if $m \times k > 80\% \times N_p$, we will reduce k by a same scaling factor for each GT to ensure at least 20% negative assignments.

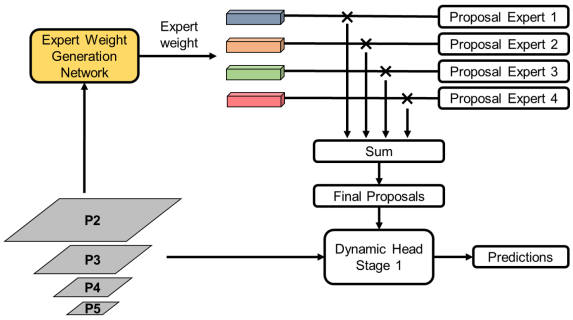
Unit Increasing Strategy. Sparse R-CNN adopts an iterative architecture to gradually increase the precision of predictions. We present a simple unit increasing strategy to promote the training of iterative structure. When the predictions of dynamic head are not correct enough in the early stage, we expect the suppliers (GT) to provide a small number of units, which constrain the matching to be stricter. When the predictions of dynamic head become more correct in the later stage, we gradually relax the constraints to let the suppliers (GT) provide a larger number of units for matching. The simple unit increasing strategy can be defined as follows.

$$k^* = k - 0.5 * (T - t), t = 1, 2, ...T \tag{2}$$

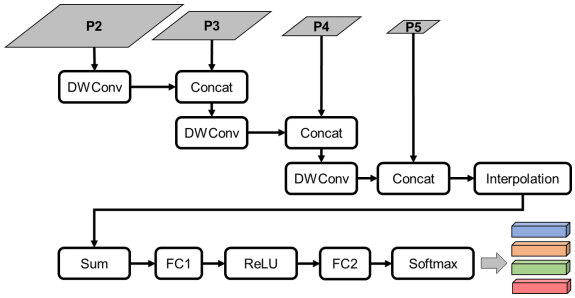
where we use the default number of iteration stages ($T = 6$) in our method.

3.3. Dynamic Proposal Generation

In Sparse R-CNN, a set of N_p proposal boxes and N_p proposal features are fed into the dynamic head together with the features extracted from the FPN backbone (P_2 to P_5). These proposals are learnable during training but fixed for different images during inference. Motivated by dynamic convolution, we propose to generate dynamic proposal boxes and features with respect to the input image to improve performance. In our design (Figure 3 (a)), proposal



(a) Dynamic Proposal Generation



(b) Staircase Structure

Figure 3. Illustrations of the proposed (a) Dynamic Proposal Generation (DPG) module and (b) staircase structure in DPG to produce expert weights.

boxes / features are a linear combination of N_e distinct sets of proposal boxes / features, and each set is referred to as an *expert*. The coefficients (referred to as expert weights) to combine the experts are generated by an expert weight generation network (Figure 3 (b)). Our DPG module can be formulated as follows.

$$\begin{aligned} \mathcal{P}_o^b &= \textstyle\sum_{i=1}^{N_e} \mathcal{P}_i^b * W_i \\ \mathcal{P}_o^f &= \textstyle\sum_{i=1}^{N_e} \mathcal{P}_i^f * W_i \\ (W_1, W_2, ..., W_{N_e}) &= G(\mathcal{F}) \end{aligned} \tag{3}$$

where $\mathcal{P}_i^b$ indicates the output dynamic proposal boxes, $\mathcal{P}_i^f$ indicates the output dynamic proposal features, W_i is the proposal expert weight learned by the expert weight generation network G, $\mathcal{F}$ indicates the features extracted from the FPN backbone (P_2 to P_5).

Staircase Structure. Our expert weight generation network follows the basic design of dynamic convolution structure, as shown in Figure 3 (b). We also use the temperature annealing operation (tao) in softmax to control the expert weights and make the training process more effective. We build a staircase architecture to aggregate the features from different pyramid levels. The P_2 to P_5 features descend in scale: the width and height of P_i is $1/2$ of that of P_{i-1}. Depth-wise convolution with 3×3 kernel and stride of 2 is applied to the concatenation of P_i and the output by the previous level, which keeps the number of channels and downscales the intermediate features. Finally, the concatenated data is interpolated into a $4C \times 30 \times 30$ feature map ($C = 256$ for each pyramid level). Then, the $4C$ channels are fused by summation and the resulting 30×30 feature map is flattened to two FC layers. The size of the first FC is 900×1500 and the second is $1500 \times (N_e N_p)$. We build $N_e = 4$ experts and use $N_p = 300$ proposal boxes / features in our method.

All experts as well as the expert weight generation network are trained. During inference, the weight generation network takes the FPN features as input and generates the weight for each expert. Then the final proposal boxes and features are obtained by linear combinations of experts.

4. Experiments

Datasets. All experiments are conducted on the COCO 2017 dataset [16]. The training split contains about 118k samples and the validation split contains about 5k samples. This dataset labels 80 different categories of objects, which are collected from natural scenarios. We use the standard MS COCO AP as the main evaluation criterion.

Training Details. The basic training setting follows Sparse R-CNN. We use the pre-trained networks (e.g., ResNet-50 [11]) on ImageNet [5] with 5 FPN levels as backbone. During training, we use AdamW optimizer [19] and the weight decay is set to 0.0001. We train the model with batch size of 16 for 36 epochs. The initial learning rate is 2.5×10^{-5} and scaled with 0.1 at 27-th and 33-th epoch. Xavier initialization [10] is applied to newly added layers. We follow Sparse R-CNN to adopt the same multi-scale training procedure by resizing the input images such that the shortest side is at least 480 and at most 800 pixels while the longest at most 1333. Following Sparse R-CNN, we adopt the iterative structure with 6 stages for training. Our experiments are conducted on 4 Nvidia A100 GPUs and training of Dynamic Sparse R-CNN takes around 37 hours with the ResNet-50 backbone.

Inference Details. For inference, 300 boxes and the associated scores are output as predictions. The score of each box is the probability that the box contains an object. No post-processing on these boxes is needed during inference. In our dynamic label assignment based on OTA, non-maximum suppression (NMS) is applied with threshold of 0.7.

4.1. Comparisons to the State-of-the-Arts

Comparisons to Transformer-based Detectors. Table 1 compares our Dynamic Sparse R-CNN with the state-of-the-art Transformer-based object detection methods which

Methods	Backbone	Train Epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
CNN-based Detectors:								
Faster R-CNN [30]	ResNet-50	36	40.2	61.0	43.8	24.2	43.5	52.0
Faster R-CNN [30]	ResNet-101	36	42.0	62.5	45.9	25.2	45.6	54.6
RetinaNet [30]	ResNet-50	36	38.7	58.0	41.5	23.3	42.3	50.3
RetinaNet [30]	ResNet-101	36	40.4	60.2	43.2	24.0	44.3	52.2
Cascade R-CNN [30]	ResNet-50	36	44.3	62.2	48.0	26.6	47.7	57.7
ATSS [32]	ResNet-101	24	43.5	-	-	-	-	-
PAA [12]	ResNet-101	24	44.6	-	-	-	-	-
OTA [8]	ResNet-50	12	40.7	58.4	44.3	23.2	45.0	53.6
Transformer-based Detectors:								
DETR [2]	ResNet-50	500	42.0	62.4	44.2	20.5	45.8	61.1
DETR [2]	ResNet-101	500	43.5	63.8	46.4	21.9	48.0	61.8
DETR [2]	ResNet-101-DC5	500	44.9	64.7	47.7	23.7	49.5	62.3
Conditional DETR [20]	ResNet-50	108	43.0	64.0	45.7	22.7	46.7	61.5
Conditional DETR [20]	ResNet-101	108	44.5	65.6	47.5	23.6	48.4	63.6
Conditional DETR [20]	ResNet-101-DC5	108	45.9	66.8	49.5	27.2	50.3	63.3
Anchor DETR [29]	ResNet-50	50	42.1	63.1	44.9	22.3	46.2	60.0
Anchor DETR [29]	ResNet-101	50	43.5	64.3	46.6	23.2	47.7	61.4
Anchor DETR [29]	ResNet-101-DC5	50	45.1	65.7	48.8	25.8	49.4	61.6
Sparse_R-CNN [25]	ResNet-50	36	45.0	63.4	48.2	26.9	47.2	59.5
Sparse_R-CNN [25]	ResNet-101	36	46.4	64.6	49.5	28.3	48.3	61.6
TSP-RCNN [26]	ResNet-50	96	45.0	64.5	49.6	29.7	47.7	58.0
TSP-RCNN [26]	ResNet-101	96	46.5	66.0	51.2	29.9	49.7	59.2
Ours:								
Dynamic Sparse R-CNN	ResNet-50	36	47.2	66.5	51.2	30.1	50.4	61.7
Dynamic Sparse R-CNN	ResNet-101	36	47.8	67.0	52.0	31.0	51.1	62.2

Table 1. Detection performance comparisons (%) on the COCO 2017 validation set.

Setting	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
Baseline	45.0	63.4	48.2	26.9	47.2	59.5
+ DPG, w/o staircase	45.3	63.2	49.5	28.8	48.2	59.1
+ DPG, w/ staircase	45.7	63.9	50.0	28.8	48.2	59.8
+ DPG, + DLA, dynamic q=8, w/o unit increasing strategy	46.0	65.0	49.9	28.7	49.2	61.1
+ DPG, + DLA, dynamic q=8, w/ unit increasing strategy	47.2	66.5	51.3	30.1	50.4	61.7

Table 2. Effect of each algorithmic component of our method.

are mostly related to our method. The results show that Dynamic Sparse R-CNN outperforms not only the original Sparse R-CNN, but also the other improved DETR methods, such as Conditional DETR and Anchor DETR. For example, with the same ResNet-50 backbone, our work surpasses Conditional DETR by 4.2% AP and Anchor DETR by 5.1% AP. Equipped with a larger ResNet-101 backbone, we also obtain improved performance compared to prior methods by a large margin. On the other hand, we only train the network for 36 epochs (same as the Sparse R-CNN baseline), which are significantly shorter than other Transformer-based detectors. We also evaluate our method on the COCO test-dev set. Our Dynamic Sparse R-CNN achieves 47.2% AP with ResNet-50 and 47.9% with ResNet-101, which surpasses TSP-RCNN with ResNet-101 (46.6%).

Comparisons to CNN-based Detectors. We also compare our Dynamic Sparse R-CNN with the state-of-the-art CNN-based methods. With the same 3× training scheduler (i.e., 36 epochs), our method outperforms Faster R-CNN, RetinaNet and Cascade R-CNN. The methods of ATSS, PAA and OTA also explore improved many-to-one label assignment schemes which are related to our DLA. Our Dynamic Sparse R-CNN obtain superior performance com-

Backbone	Matcher	unit	loss	unit increasing strategy	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
R50	Hungarian	✗	✗	✗	45.0	63.4	48.2	26.9	47.2	59.5
R50	OTA	fixed k=1	✗	✗	44.7	64.9	48.0	28.2	46.9	59.3
R50	OTA	fixed k=2	✗	✗	45.9	65.1	49.8	28.8	48.6	60.9
R50	OTA	fixed k=3	✗	✗	45.9	65.2	50.0	28.6	48.6	61.0
R50	OTA	dynamic q=8	✗	✗	46.1	64.6	50.1	27.9	49.2	61.9
R50	OTA	dynamic q=8	two losses	✗	46.1	65.2	50.0	29.4	49.7	60.9
R50	OTA	dynamic q=8	two losses	✓	46.7	65.9	50.9	29.8	49.8	61.3

Table 3. Effect of different matchers. Dynamic proposal generation is not used in this ablation experiment.

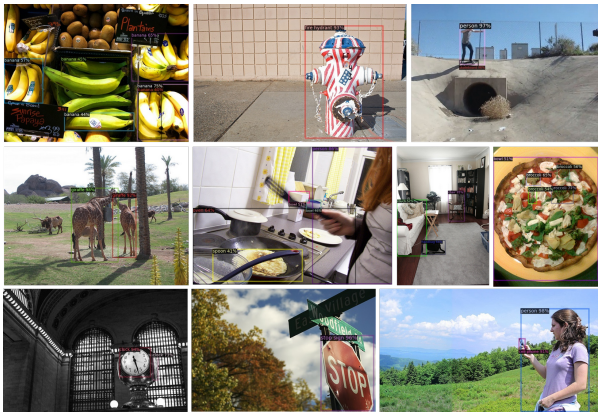


Figure 4. Visualization of sampled detection results by Dynamic Sparse R-CNN with the ResNet-50 backbone.

pared to these methods with the same backbone, e.g., surpassing OTA by 6.5% AP with ResNet-50 and PAA by 3.2% AP with ResNet-101 on the COCO validation set.

Qualitative Results. Figure 4 visualizes sampled detection results by our Dynamic Sparse R-CNN. Our method can detect objects correctly with varying scales, appearances, etc.

4.2. Ablation Study

Contributions from Algorithmic Components. We conduct ablation experiments to examine the contributions from each algorithmic components. As shown in Table 2, the dynamic proposal generation design boosts AP by 0.7 points with the staircase structure to aggregate features from multiple pyramid levels. In particular, both the AP_{75} and AP_s values witness an enhance for nearly 2 points, demonstrating that that DPG helps the model to perform better in a more strict IoU criterion and detecting small objects. The intuition behind this improvement is that the DPG helps to provide a more diverse range of proposal boxes and features to the dynamic head for better predictions. Our staircase structure can better utilize the FPN features for generating expert weights. Without staircase structure, the FPN features are directly interpolated into the 30×30 feature maps and concatenated to be fed into the first FC layer. The results show that this staircase structure brings 0.4% AP gain. By applying many-to-one label assignment based on OTA, we can boost the performance from 45.7% to 46.0%. In this setting, the units are set based on the same Dynamic k Estimation method for all the iteration stages. We find that our simple unit increasing strategy can further improve the performance, reaching 47.2% AP with a single model. These results demonstrate the effectivenss of our designs of DLA and DPG.

q	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
4	46.7	66.0	51.1	31.5	50.1	60.5
5	46.7	66.2	50.9	30.6	49.8	61.1
6	46.7	66.0	50.9	30.2	50.0	60.7
7	46.4	65.7	50.4	30.2	49.5	60.7
8	47.2	66.5	51.3	30.1	50.4	61.7
9	46.1	65.2	50.1	29.0	49.5	60.6

Table 4. Effect of q in Dynamic k Estimation with unit increasing strategy and dynamic proposal generation.

Effect of Different Matchers. As shown in Table 3, OTA matchers with fixed k values ($k = 2, 3$) gives a 0.9-point lift of AP compared to the baseline. The OTA matcher with $q = 8$ in Dynamic k Estimation brings a higher increase of 1.1 points, which demonstrates the effectiveness of using dynamic k. The unit increasing strategy further enhances AP to 46.7%, indicating that this simple design is effective. In addition, the OTA matcher with $q = 8$ and the unit increasing strategy brings a nearly 3-point increase in terms of both AP_{75} and AP_s. The intuition behind the significant increase is that our dynamic many-to-one matching scheme produces more diverse options of prediction boxes to match a ground truth. This scheme especially favors the detection of small objects.

Effect of q. As shown in Table 4, we try different choices of q in Dynamic k Estimation and find $q = 8$ works best. It

#Experts	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
3	45.4	63.4	50.0	28.6	48.4	59.6
4	45.7	63.9	50.0	28.8	48.2	59.8
5	45.3	63.2	49.5	27.6	47.8	60.0

Table 5. Effect of the number of experts. Dynamic label assignment is not used in this ablation experiment.

is noted that all the results in Table 4 outperforms the one-to-one matching baseline (45.0%), which validate the effectiveness of our dynamic many-to-one matching scheme.

Effect of Number of Experts. As shown in Table 5 , we try different numbers of experts and use 4 experts as default in our method.

5. More Analysis

Figure 5 compares the detailed training curve of the AP values between Sparse R-CNN and Dynamic Sparse R-CNN. We observe that our Dynamic Sparse R-CNN outperforms the baseline throughout the training iterations. The results further validate the non-trivial design of DLA and DPG.

Figure 6 compares the per-stage AP values between Sparse R-CNN and Dynamic Sparse R-CNN. The AP value of each stage is improved by at least 2 points using our method. This indicates that DLA and DPG actually contribute to the training of each iteration stage. We note that DPG is imposed for the first stage only, it helps produce better initial proposal boxes and features and could benefit the consecutive stages. Moreover, we find that Dynamic Sparse R-CNN already can achieve 46.4% AP using 4 stages, outperforming the baseline (45.0%) using 6 stages. The results show that our method can accelerate the convergence in the iterative structure.

6. Limitations

The parameter size and computation cost of our detector are slightly larger than the Sparse R-CNN baseline. Sparse R-CNN has 77.8M parameters and costs 23.28 GFLOPs, while our Dynamic Sparse R-CNN has 81.0M parameters and costs 23.30 GFLOPs. It indicates that our expert weight generation network just introduces marginal memory and computation overhead. Our Dynamic Sparse R-CNN takes 37 hours on 4 A100 GPUs for training, whereas Sparse R-CNN takes 29 hours on the same devices. The training time could be optimized further.

7. Conclusion

In this work, we propose Dynamic Sparse R-CNN by introducing two dynamic designs to improve Sparse R-CNN.

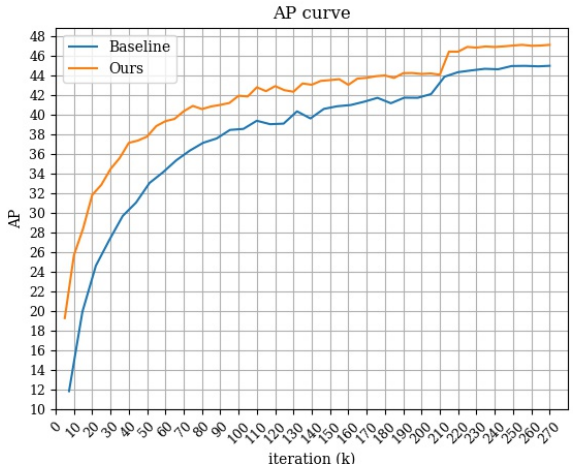


Figure 5. Comparisons of AP curves between Sparse R-CNN and Dynamic Sparse R-CNN.

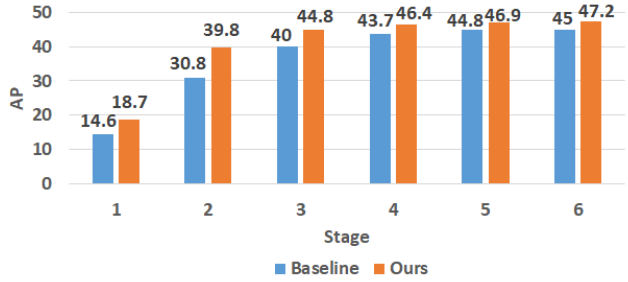


Figure 6. Comparisons of per-stage results between Sparse R-CNN and Dynamic Sparse R-CNN.

We point out that one-to-one label assignment method is sub-optimal for matching between object queries and ground truths in Transformer-based detectors. Based on optimal transport algorithm, we implement many-to-one label assignment and design a simple but effective unit increasing strategy for performance boost. We also propose a dynamic proposal generation mechanism to aggregate multiple learned experts to derive better initial proposal boxes and features. Such mechanism is motivated by dynamic convolution and produces dynamic input-dependent proposals for better detection performance. Our Dynamic Sparse R-CNN is well-motivated and reaches the state-of-the-art 47.2% AP with ResNet-50 on COCO. We expect our method can inspire new insights for object detection and consider applying our idea to more Transformer-based detectors as future work.

References

[1] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018. 1

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229. Springer, 2020. 1, 2, 3, 6

[3] Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11030–11039, 2020. 2, 3

[4] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. 2016. 1, 2

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 5

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 1

[7] Yoav Freund, Raj Iyer, Robert E Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *Journal of machine learning research*, 4(Nov):933–969, 2003. 1

[8] Zheng Ge, Songtao Liu, Zeming Li, Osamu Yoshie, and Jian Sun. Ota: Optimal transport assignment for object detection. In *CVPR*, pages 303–312, 2021. 2, 3, 4, 6

[9] Ross Girshick. Fast r-cnn. In *ICCV*, 2015. 1

[10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010. 5

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 5

[12] Kang Kim and Hee Seok Lee. Probabilistic anchor assignment with iou prediction for object detection. In *ECCV*, pages 355–371. Springer, 2020. 2, 6

[13] Tao Kong, Fuchun Sun, Huaping Liu, Yuning Jiang, Lei Li, and Jianbo Shi. Foveabox: Beyound anchor-based object detection. *TIP*, 29:7389–7398, 2020. 1, 2

[14] Zeming Li, Chao Peng, Gang Yu, Xiangyu Zhang, Yangdong Deng, and Jian Sun. Light-head r-cnn: In defense of two-stage object detector. *arXiv preprint arXiv:1711.07264*, 2017. 1, 2

[15] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 1, 2

[16] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 5

[17] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37. Springer, 2016. 1, 2

[18] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *ICCV*, 2021. 1

[19] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 5

[20] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *ICCV*, 2021. 2, 3, 6

[21] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *CVPR*, 2017. 2

[22] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 1, 2

[23] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. 39(6):1137–1149, 2015. 1, 2

[24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1

[25] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, et al. Sparse r-cnn: End-to-end object detection with learnable proposals. In *CVPR*, pages 14454–14463, 2021. 1, 2, 3, 6

[26] Zhiqing Sun, Shengcao Cao, Yiming Yang, and Kris M Kitani. Rethinking transformer-based set prediction for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3611–3620, 2021. 6

[27] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, June 2016. 1

[28] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, 2019. 1, 2

[29] Yingming Wang, Xiangyu Zhang, Tong Yang, and Jian Sun. Anchor detr: Query design for transformer-based detector. *arXiv preprint arXiv:2109.07107*, 2021. 2, 3, 6

[30] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 6

[31] Brandon Yang, Gabriel Bender, Quoc V Le, and Jiquan Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. In *NeurIPS*, volume 32, 2019. 3

[32] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *CVPR*, 2020. 2, 6

[33] Y. Zhang, J. Zhang, Q. Wang, and Z. Zhong. Dynet: Dynamic convolution for accelerating convolutional neural networks. *arXiv preprint arXiv:2004.10694*. 3

[34] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 2

[35] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 2, 3