DN-DETR: Accelerate DETR Training by Introducing Query DeNoising

Feng Li[1,2*†], Hao Zhang[1,2*†], Shilong Liu[2,3†], Jian Guo[2], Lionel M. Ni[1,4], Lei Zhang[2‡]
[1]The Hong Kong University of Science and Technology.
[2]International Digital Economy Academy (IDEA).
[3]Tsinghua University.
[4]The Hong Kong University of Science and Technology (Guangzhou).
{fliay,hzhangcx}@connect.ust.hk
{liusl20}@mails.tsinghua.edu.cn
{ni}@ust.hk
{guojian,leizhang}@idea.edu.cn

Abstract

We present in this paper a novel denoising training method to speedup DETR (DEtection TRansformer) training and offer a deepened understanding of the slow convergence issue of DETR-like methods. We show that the slow convergence results from the instability of bipartite graph matching which causes inconsistent optimization goals in early training stages. To address this issue, except for the Hungarian loss, our method additionally feeds ground-truth bounding boxes with noises into Transformer decoder and trains the model to reconstruct the original boxes, which effectively reduces the bipartite graph matching difficulty and leads to a faster convergence. Our method is universal and can be easily plugged into any DETR-like methods by adding dozens of lines of code to achieve a remarkable improvement. As a result, our DN-DETR results in a remarkable improvement (+1.9AP) under the same setting and achieves the best result (AP 43.4 and 48.6 with 12 and 50 epochs of training respectively) among DETR-like methods with ResNet-50 backbone. Compared with the baseline under the same setting, DN-DETR achieves comparable performance with 50% training epochs. Code is available at https://github.com/FengLi-ust/DN-DETR.

1. Introduction

Object detection is a fundamental task in computer vision which aims to predict the bounding boxes and classes of objects in an image. While having made remarkable progress, classical detectors [14, 15] were mainly based on convolutional neural networks, until Carion *et al*. [1] recently introduced Transformers [17] into object detection and proposed DETR (DEtection TRansformer).

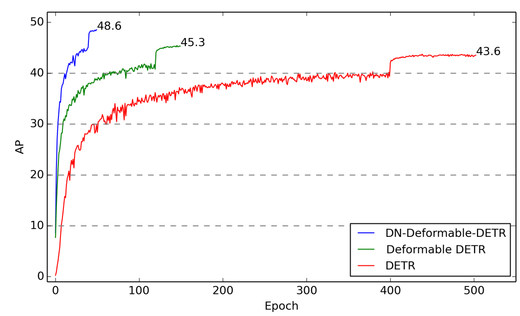

Figure 1. Convergence curve between our model DN-Deformable-DETR built upon Deformable DETR with denoising training and previous models under ResNet-50 backbone.

In contrast to previous detectors, DETR uses learnable queries to probe image features from the output of Transformer encoders and bipartite graph matching to perform set-based box prediction. Such a design effectively eliminates hand-designed anchors and non-maximum suppression (NMS) and makes object detection end-to-end optimizable. However, DETR suffers from prohibitively slow training convergence compared with previous detectors. To obtain a good performance, it usually takes 500 epochs of training on the COCO detection dataset, in contrast to 12 epochs used in the original Faster-RCNN training.

Much work [3, 11, 12, 16, 18, 20] has tried to identify the root cause and mitigate the slow convergence issue. Some of them address the problem through improving the model architecture. For example, Sun *et al*. [16] attribute the slow convergence issue to the low efficiency

*Indicates equal contribution.
†This work was done when Feng Li, Hao Zhang, and Shilong Liu were interns at IDEA.
‡Corresponding author.

of the cross-attention and proposed an encoder-only DETR. Dai *et al*. [3] designed a ROI-based dynamic decoder to help the decoder focus on regions of interest. More recent works propose to associate each DETR query with a specific spatial position rather than multiple positions for more efficient feature probing [11, 12, 18, 20]. For instance, Conditional DETR [12] decouples each query into a content part and a positional part, enforcing a query to have a clear correspondence with a specific spatial position. Deformable DETR [20] and Anchor DETR [18] directly treat $2D$ reference points as queries to perform cross-attention. DAB-DETR [11] interprets queries as 4-D anchor boxes and learns to progressively improve them layer by layer.

Despite all the progress, few work pays attention to the bipartite graph matching part for more efficient training. In this study, we find that the slow convergence issue also results from the discrete bipartite graph matching component, which is unstable especially in the early stages of training due to the nature of stochastic optimization. As a consequence, for the same image, a query is often matched with different objects in different epochs, which makes optimization ambiguous and inconstant.

To address this problem, we propose a novel training method by introducing a query denoising task to help stabilize bipartite graph matching in the training process. Since previous works have shown effective to interpret queries as reference points [18, 20] or anchor boxes [11] which contain positional information, we follow their viewpoint and use 4D anchor boxes as queries. Our solution is to feed noised ground truth bounding boxes as noised queries together with learnable anchor queries into Transformer decoders. Both kinds of queries have the same input format of (x, y, w, h) and can be fed into Transformer decoders simultaneously. For noised queries, we perform a denoising task to reconstruct their corresponding ground truth boxes. For other learnable anchor queries, we use the same training loss including bipartite matching as in the vanilla DETR. As the noised bounding boxes do not need to go through the bipartite graph matching component, the denoising task can be regarded as an easier auxiliary task, helping DETR alleviate the unstable discrete bipartite matching and learn bounding box prediction more quickly. Meanwhile, the denoising task also helps lower the optimization difficulty because the added random noise is usually small. To maximize the potential of this auxiliary task, we also regard each decoder query as a bounding box + a class label embedding so that we are able to conduct both box denoising and label denoising.

In summary, our method is a denoising training approach. Our loss function consists of two components. One is a reconstruction loss and the other is a Hungarian loss which is the same as in other DETR-like methods. Our method can be easily plugged into any existing DETR-like method. For convenience, we utilize DAB-DETR [11] to evaluate our method since their decoder queries are explicitly formulated as 4D anchor boxes (x, y, w, h). For DETR variants that only support 2D anchor points such as anchor DETR [18], we can do denoising on anchor points. For those that do not support anchors like the vanilla DETR [1], we can do linear transformation to map 4D anchor boxes to the same latent space as for other learnable queries.

To the best of our knowledge, this is the first work to introduce the denoising principle into detection models. We summarize our contribution as follows:

1. We design a novel training method to speedup DETR training. Experimental results show that our method not only accelerates training convergence, but also leads to a remarkably better training result — achieve the best result among all detection algorithms in the 12-epoch setting. Moreover, our method shows a remarkable improvement (+**1.9** AP) over our baseline DAB-DETR and can be easily integrated into other DETR-like methods.

2. We analyze the slow convergence of DETR from a novel viewpoint and give a deeper understanding of DETR training. We design a metric to evaluate the instability of bipartite matching and verify that our method can effectively lower the instability.

3. We conduct a series of ablation studies to analyze the effectiveness of different components of our model such as noise, label embedding, and attention mask.

2. Related Work

Classical CNN-based detectors can be divided into 2 categories, one-stage and two-stage methods. Two-stage methods [6, 7] first generate some region proposals and then decide whether each region contains an object and do bounding box regression to get a refined box. Ren *et al*. [15] proposed an end-to-end method which utilizes a Region Proposal Network to predict anchor boxes. In contrast to two-stage methods, one-stage methods [13, 14] directly predict the offset of real boxes relative to anchor boxes. Overall, they are all anchor-based methods.

Carion *et al*. [1] proposed an end-to-end object detector based on Transformers [17] named DETR (DEtection TRansformer) without using anchors. While DETR achieves comparable results with Faster-RCNN [15], its training suffers severely from the slow convergence problem — it needs 500 epochs of training to obtain a good performance.

Many recent works have attempted to speedup the training process of DETR. Some find the cross attention of Transformer decoders in DETR inefficient and make improvement from different ways. For example, Dai *et al*. [3]

designed a dynamic decoder that can focus on regions of intersts from a coarse-to-fine manner and lower the learning difficulty. Sun *et al*. [16] discarded the Transformer decoder and proposed an encoder only DETR. Another series of works make improvements in decoder queries. Zhu *et al*. [20] designed an attention module that only attend to some sampling points around a reference point. Meng et al. [12] decoupled each decoder query into a content part and a position part and only utilized the content-to-content and position-to-position terms in the cross-attention formulation. Yao *et al*. [19] utilized a Region Proposal Network (RPN) to propose top-K anchor points. DAB-DETR [11] uses 4-D box coordinates as queries and updates boxes layer-by-layer in a cascade manner.

Despite all the progress, none of them treats bipartite graph matching used in the Hungarian loss as a main reason for slow convergence. Sun *et al*. [16] analyzed the impact of Hungarian loss by using a pre-trained DETR as a teacher to provide the ground-truth label assignment for a student model and train the student model. They found that the label assignment only helps the convergence in the early stage of training but does not influence the final performance significantly. Therefore, they concluded that the Hungarian loss is not a main reason for slow convergence. In this work, we give a different analysis with an effective solution that leads to a different conclusion.

We adopt DAB-DETR as the detection architecture to evaluate our training method, where the label embedding appended with an indicator is used to replace the decoder embedding part to support label denoising. The difference between our method and other methods is mainly in the training method. In addition to the Hungarian loss, we add a denoising loss as an easier auxiliary task that can accelerate training and boost the performance significantly. Chen *et al*. [2] augments their sequence with synthetic noise objects, but is totally different from our method. They set the targets of noise objects to ”noise” class (not belonging to any ground truth classes) so that they can delay the End-of-Sentence (EOS) token and improve the recall. In contrast to their method, we set the target of noised boxes to the original boxes and the motivation is to bypass bipartite graph matching and directly learn to approximate ground truth boxes.

3. Why Denoising accelerates DETR training?

Hungarian matching is a popular algorithm in graph matching. Given a cost matrix, the algorithm outputs an optimal matching result. DETR is the first algorithm that adopts Hungarian matching in object detection to solve the matching problem between predicted objects and ground truth objects. DETR turns ground truth assignment to a dynamic process, which brings in an instability problem due to its discrete bipartite matching and the stochastic training process. There are works [5] showing that Hungarian matching does not result in a stable matching since blocking pairs exist. A small change of the cost matrix may cause an enormous change in the matching result, which will further lead to inconsistent optimization goals for decoder queries.

We view the training process of DETR-like models as two stages, learning “good anchors” and learning relative offsets. Decoder queries are responsible for learning anchors as shown in previous works [11,20]. The inconsistent update of anchors can make it difficult to learn relative offsets. Therefore, in our method, we leverage a denoising task as a training shortcut to make relative offset learning easier, as the denoising task bypasses bipartite matching. Since we interpret each decoder query as a 4-D anchor boxes, a noised query can be regarded as a “good anchor” which has a corresponding ground truth box nearby. The denoising training thus has a clear optimization goal - to predict the original bounding box, which essentially avoids the ambiguity brought by Hungarian matching.

To quantitatively evaluate the instability of the bipartite matching result, we design a metric as follows. For a training image, we denote the predicted objects from Transformer decoders as $\mathbf{O^i} = \left\{O_0^i, O_1^i, ..., O_{N-1}^i\right\}$ in the i-th epoch, where N is the number of predicted objects, and the ground truth objects as $\mathbf{T} = \{T_0, T_1, T_2, ..., T_{M-1}\}$ where M is the number of ground truth objects. After bipartite matching, we compute an index vector $\mathbf{V^i} = \left\{V_0^i, V_1^i, ..., V_{N-1}^i\right\}$ to store the matching result of epoch i as follows.

$$V_n^i = \begin{cases} m, & \text{if } O_n^i \text{ matches } T_m \\ -1, & \text{if } O_n^i \text{ matches nothing} \end{cases} \tag{1}$$

We define the instability of epoch i for one training image

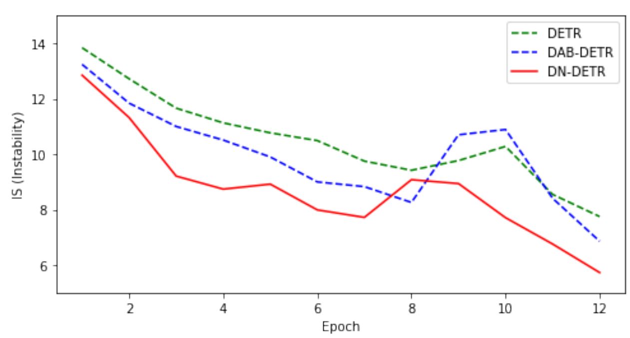

Figure 2. The IS of DAB-DETR and DN-DETR during training. For each method, we train 12 epoch on the same setting. We test the change of the Hungarian matching between each two epochs on the Validation set as the IS.

as the difference between its V^i and V^{i-1}, which is calculated as

$$IS^i = \sum_{j=0}^{N} \mathbb{1}(V_n^i \neq V_n^{i-1}) \tag{2}$$

where $\mathbb{1}(\cdot)$ is the indicator function. $\mathbb{1}(x) = 1$ if x is true and 0 otherwise. The instability of epoch i for the whole data set is averaged over the instability numbers for all images. We omit the index for an image for notation simplicity in Eq. (1) and Eq. (2).

Fig. 2 shows a comparison of IS between our DN-DETR (DeNoising DETR) and DAB-DETR. We conduct this evalutaion on COCO 2017 validation set [10] which has 7.36 objects per image on average. So the largest possible IS is $7.36 \times 2 = 14.72$. Fig. 2 clearly shows that our method effectively alleviates the instability of matching.

4. DN-DETR

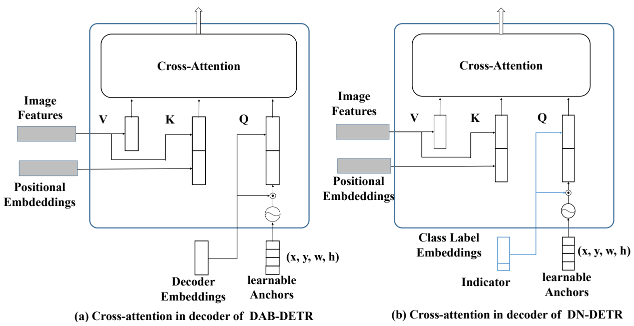

Figure 3. Comparison of the cross-attention part DAB-DETR and our DN-DETR (a)DAB-DETR directly uses dynamically updated anchor boxes to provide both a reference query point (x, y) and a reference anchor size (w, h) to improve the cross-attention computation. (b) DN-DETR specify the decoder embedding as label embedding and add an indicator to differentiate denoising task and matching task.

4.1. Overview

We base on the architecture of DAB-DETR [11] to implement our training method. Similar to DAB-DETR, we explicitly formulates the decoder queries as box coordinates. The only difference between our architecture and theirs lies in the decoder embedding, which is specified as class label embedding to support label denoising. Our main contribution is the training method as shown in Fig. 4.

Similar to DETR, our architecture contains a Transformer encoder and a Transformer decoder. On the encoder side, the image features are extracted with a CNN backbone and then fed into the Transformer encoder with positional encodings to attain refined image features. On the decoder side, queries are fed into the decoder to search for objects through cross attention.

We denote decoder queries as $\mathbf{q} = \{q_0, q_1, ..., q_{N-1}\}$ and the output of the Transformer decoder as $\mathbf{o} = \{o_0, o_1, ..., o_{N-1}\}$. We also use F and A to denote the refined image features after the Transformer encoder and the attention mask derived based on the denoising task design. We can formulate our method as follows.

$$\mathbf{o} = D(\mathbf{q}, F|A) \tag{3}$$

where D denotes the Transformer decoder.

There are two parts of decoder queries. One is the matching part. The inputs of this part are learnable anchors, which are treated in the same way as in DETR. That is, the matching part adopts bipartite graph matching and learns to approximate the ground truth box-label pairs with matched decoder outputs. The other is the denoising part. The inputs of this part are noised ground-truth (GT) box-label pairs which are called **GT objects** in rest of the paper. The outputs of the denoising part aims to reconstruct GT objects.

In the following, we abuse the notations to denote the denoising part as $\mathbf{q} = \{q_0, q_1, ..., q_{K-1}\}$ and the matching part as $\mathbf{Q} = \{Q_0, Q_1, ..., Q_{L-1}\}$. So the formulation of our method becomes

$$\mathbf{o} = D(\mathbf{q}, \mathbf{Q}, F|A) \tag{4}$$

To increase the denoising efficiency, we propose to use multiple versions of noised GT objects in the denoising part. Further more, we utilize an attention mask to prevent information leakage from the denoising part to the matching part and among different noised versions of the same GT object.

4.2. Intro to DAB-DETR

Many recent works associate DETR queries with different positional information. DAB-DETR follows this analysis and explicitly formulates each query as 4D anchor coordinates. As shown in Fig. 3(a), a query is specified as a tuple (x, y, w, h), where x, y are the center coordinates and w, h are the corresponding width and height of each box. In addition, the anchor coordinates are dynamically updated layer by layer. The output of each decoder layer contains a tuple $(\Delta x, \Delta y, \Delta w, \Delta h)$ and the anchor is updated to $(x + \Delta x, y + \Delta y, w + \Delta w, h + \Delta h)$.

Note that our proposed method is mainly a training method which can be integrated into any DETR-like models. To test on DAB-DETR, we only add minimal modifications: specifying the decoder embedding as label embedding, as shown in Fig. 3(b).

4.3. Denoising

For each image, we collect all GT objects and add random noises to both their bounding boxes and class labels. To maximize the utility of denoising learning, we use multiple noised versions for each GT object.

We consider adding noise to boxes in two ways: center shifting and box scaling. We define λ_1 and λ_2 as the noise scale of these 2 noises. For center shifting, we add a random noise $(\Delta x, \Delta y)$, to the box center and make sure that $|\Delta x| < \frac{\lambda_1 w}{2}$ and $|\Delta y| < \frac{\lambda_1 h}{2}$, where $\lambda_1 \in (0, 1)$ so that the center of the noised box will still lie inside the

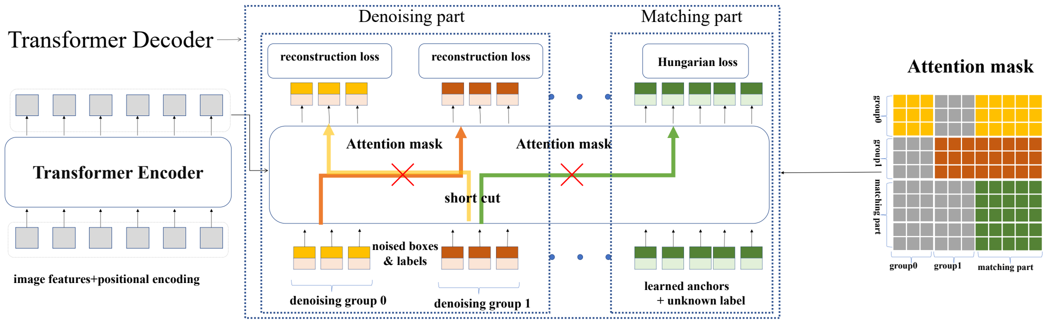

Figure 4. The overview of our training method. There are two parts of queries, namely the denoising part and the matching part. The denoising part contains ≥ 1 denoising groups. The attention masks from the matching part to the denoising part and among denoising groups are set to 1 (block) to block information leakage. In the figure, the yellow, brown and green grids in the attention mask represent 0 (unblock) and grey grids represent 1 (block).

original bounding box. For box scaling, we set a hyper-parameter $\lambda_2 \in (0, 1)$. The width and height of the box are randomly sampled in $[(1-\lambda_2)w, (1+\lambda_2)w]$ and $[(1-\lambda_2)h, (1+\lambda_2)h]$, respectively.

For label noising, we adopt label flipping, which means we randomly flip some ground-truth labels to other labels. Label flipping forces the model to predict the ground-truth labels according to the noised boxes to better capture label-box relationship. We have a hyper-parameter γ to control the ratio of labels to flip. The reconstruction losses are l_1 loss and GIOU loss for boxes and focal loss [9] for class labels as in DAB-DETR. We use a function $\delta(\cdot)$ to denote the the noised GT objects. Therefore, each query in the denoising part can be represented as $q_k = \delta(t_m)$ where t_m is m-th GT object.

Notice that denoising is only considered in training, during inference the denoising part is removed, leaving only the matching part.

4.4. Attention Mask

Attention mask is a component of great importance in our model. Without attention mask, the denoising training will compromise the performance instead of improving it as shown in Table 4.

To introduce attention mask, we need to first divide the noised GT objects into groups. Each group is a noised version of all GT objects. The denoising part becomes

$$\mathbf{q} = \{\mathbf{g_0}, \mathbf{g_1}, ..., \mathbf{g_{P-1}}\} \tag{5}$$

where $\mathbf{g_p}$ is defined as the p-th **denoising group**. Each denoising group contains M queries where M is the number of GT objects in the image. So we have

$$\mathbf{g_p} = \{q_0^p, q_1^p, ..., q_{M-1}^p\} \tag{6}$$

where $q_m^p = \delta(t_m)$.

The purpose of the attention mask is to prevent information leakage. There are two types of potential information leakage. One is that the matching part may see the noised GT objects and easily predict GT objects. The other is that one noised version of a GT object may see another version. Therefore, our attention mask is to make sure the matching part cannot see the denoising part and the denoising groups cannot see each other as shown in Fig. 4.

We use $\mathbf{A} = [\mathbf{a}_{ij}]_{W \times W}$ to denote the attention mask where $W = P \times M + N$. P and M are the number of groups and GT objects. N is the number of queries in the matching part. We let the first $P \times M$ rows and columns to represent the denoising part and the latter to represent the matching part. $a_{ij} = 1$ means the i-th query cannot see the j-th query and $a_{ij} = 0$ otherwise. We devise the attention mask as follows

$$a_{ij} = \begin{cases} 1, & \text{if } j < P \times M \text{ and } \lfloor \frac{i}{M} \rfloor \neq \lfloor \frac{j}{M} \rfloor; \\ 1, & \text{if } j < P \times M \text{ and } i \geq P \times M; \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Note that whether the denoising part can see the matching part or not will not influence the performance, since the queries of matching part are learned queries that contain no information of the ground truth objects.

The extra computation introduced by multiple denoising groups is negligible—when 5 denoising groups are introduced, GFLOPs for training is only increased from 94.4 to 94.6 for DAB-DETR with R50 backbone and there is no computation overhead for testing.

4.5. Label Embedding

The decoder embedding is specified as label embedding in our model to support both box denoising and label denoising. Except for the 80 classes in COCO 2017 [10], we also consider an unknown class embedding which is used in the matching part to be semantically consistent with the denoising part. We also append an indicator to label embedding. The indicator is 1 if a query belongs to the denoising part and 0 otherwise.

Model	#epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	GFLOPs	Params
DETR-R50 [1]	500	42.0	62.4	44.2	20.5	45.8	61.1	86	41M
Faster RCNN-FPN-R50 [15]	108	42.0	62.1	45.5	26.6	45.5	53.4	180	42M
Anchor DETR-R50 [18]	50	42.1	63.1	44.9	22.3	46.2	60.0	–	39M
Conditional DETR-R50 [12]	50	40.9	61.8	43.3	20.8	44.6	59.2	90	44M
DAB-DETR-R50 [11]	50	42.2	63.1	44.7	21.5	45.7	60.3	94	44M
DN-DETR-R50	50	**44.1(+1.9)**	64.4	46.7	22.9	48.0	63.4	94	44M
DETR-R101 [1]	500	43.5	63.8	46.4	21.9	48.0	61.8	152	60M
Faster RCNN-FPN-R101 [15]	108	44.0	63.9	47.8	27.2	48.1	56.0	246	60M
Anchor DETR-R101 [18]	50	43.5	64.3	46.6	23.2	47.7	61.4	–	58M
Conditional DETR-R101 [12]	50	42.8	63.7	46.0	21.7	46.6	60.9	156	63M
DAB-DETR-R101 [11]	50	43.5	63.9	46.6	23.6	47.3	61.5	174	63M
DN-DETR-R101	50	**45.2(+1.7)**	65.5	48.3	24.1	49.1	65.1	174	63M
DETR-DC5-R50 [1]	500	43.3	63.1	45.9	22.5	47.3	61.1	187	41M
Anchor DETR-DC5-R50 [18]	50	44.2	64.7	47.5	24.7	48.2	60.6	151	39M
Conditional DETR-DC5-R50 [12]	50	43.8	64.4	46.7	24.0	47.6	60.7	195	44M
DAB-DETR-DC5-R50 [11]	50	44.5	65.1	47.7	25.3	48.2	62.3	202	44M
DN-DETR-DC5-R50	50	**46.3(+1.8)**	66.4	49.7	26.7	50.0	64.3	202	44M
DETR-DC5-R101 [1]	500	44.9	64.7	47.7	23.7	49.5	62.3	253	60M
Anchor DETR-R101 [18]	50	45.1	65.7	48.8	25.8	49.4	61.6	–	58M
Conditional DETR-DC5-R101 [12]	50	45.0	65.5	48.4	26.1	48.9	62.8	262	63M
DAB-DETR-DC5-R101 [11]	50	45.8	65.9	49.3	27.0	49.8	63.8	282	63M
DN-DETR-DC5-R101	50	**47.3(+1.5)**	67.5	50.8	28.6	51.5	65.0	282	63M

Table 1. Results for our DN-DETR and other detection models under the same setting. All DETR-like models except DETR use 300 queries, while DETR uses 100.

Model	MultiScale	#epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	GFLOPs	Params
Faster R50-FPN 1x [15]	✓	12	37.9	58.8	41.1	22.4	41.1	49.1	180	40M
DETR-R50 1x [1]		12	15.5	29.4	14.5	4.3	15.1	26.7	86	41M
DAB-DETR-DC5-R50 [11]		12	38.0	60.3	39.8	19.2	40.9	55.4	216	44M
DN-DETR-DC5-R50		12	**41.7(+3.7)**	61.4	44.1	21.2	45.0	60.2	216	44M
Deformable DETR-R50 1x [20]	✓	12	37.2	55.5	40.5	21.1	40.7	50.5	173	40M
Dynamic DETR-R50† 1x (without dynamic encoder)	✓	12	40.2	58.6	43.4	−−	−	−	−	−
Dynamic DETR-R50† 1x [4]	✓	12	42.9	61.0	46.3	24.6	44.9	54.4	−	−
DN-Deformable-DETR-R50 [4]	✓	12	**43.4**	61.9	47.2	24.8	46.8	59.4	195	48M
DAB-DETR-DC5-R101 [11]		12	40.3	62.6	42.7	22.2	44.0	57.3	282	63M
DN-DETR-DC5-R101		12	**42.8(+2.5)**	62.9	45.7	23.3	46.6	61.3	282	63M
Faster R101 FPN [15]	✓	108	44.0	63.9	47.8	27.2	48.1	56.0	246	60M
DN-Deformable-DETR-R101	✓	12	**44.1**	62.8	47.9	26.0	47.8	61.3	275	67M

Table 2. Results for our DN-DETR and other detection models on the 1x setting. Superscript † indicates that we check with the authors of Dynamic DETR through private communication, their encoder deign makes their single-scale and multi-scale results almost identical. .

5. Experiment

5.1. Setup

Dataset: We show the effectiveness of DN-DETR on the challenging COCO 2017 [10] Detection task. Following the common practice, we report the standard mean average precision (AP) result on the COCO validation dataset under different IoU thresholds and object scales.

Implementation Details: We test the effectiveness of the denoising training on DAB-DETR, which is composed of a CNN backbone, multiple Transformer encoder layers and decoder layers. We also show that denoising training can be plugged into other DETR-like models to boost performance. For example, our DN-Deformable-DETR is built upon Deformable DETR in multi-scale setting.

We adopt several ResNet models [8] pre-trained on ImageNet as our backbones and report our results on 4 ResNet settings: ResNet-50 (R50), ResNet-101 (R101), and their 16×-resolution extensions ResNet-50-DC5 (DC5-R50) and ResNet-101-DC5 (DC5-R101). For hyperparameters, we follow DAB-DETR to use a 6-layer Transformer encoder and a 6-layer Transformer decoder and 256 as the hidden dimension. We add uniform noise on boxes and set the hyperparameters with respect to noise as $\lambda_1 = 0.4$, $\lambda_2 = 0.4$, and $\gamma = 0.2$. For the learning rate scheduler, we use an initial learning rate (lr) 1×10^{-4} and drop lr at the 40-th epoch by multiplying 0.1 for the 50-epoch setting and at the 11-th epoch by multiplying 0.1 for the 12-epoch setting. We use the AdamW optimizer with weight decay of 1×10^{-4} and train our model on 8 Nvidia A100 GPUs. The batch size is

Model	MultiScale	#epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	GFLOPs	Params
Deformable DETR-R50 [20]	✓	50	43.8	62.6	47.7	26.4	47.1	58.0	173	40M
SMCA-R50 [6]	✓	50	43.7	63.6	47.2	24.2	47.0	60.4	152	40M
TSP-RCNN-R50 [16]	✓	96	45.0	64.5	49.6	29.7	47.7	58.0	188	–
Dynamic DETR-R50* [4]	✓	50	47.2	65.9	51.1	28.6	49.3	59.1	–	–
DAB-Deformable-DETR-R50	✓	50	46.9	66.0	50.8	30.1	50.4	62.5	195	48M
DN-Deformable-DETR-R50	✓	50	**48.6**	67.4	52.7	31.0	52.0	63.7	195	48M

Table 3. Best results for our DN-DETR and other detection models with the ResNet-50 backbone. * indicates it is the test-dev result.

16. Unless otherwise specified, we use 5 denoising groups.

We conduct a series of experiments to demonstrate the performance improvement as shown in Table 1, where we follow the basic settings in DAB-DETR without any bells and whistles in training. To compare with the state-of-the-art performance in the 12 epoch setting (the so called $1\times$ setting on Detectron2) and the standard 50 epoch setting (most widely used in DETR-like models) in Table 2 and 3, we follow DAB-DETR to use 3 pattern embeddings as in Anchor DETR [18]. All our comparisons with DAB-DETR and its variants are under exactly the same setting.

DN-Deformable-DETR: To show the effectiveness of denoising training applied in other DETR-like models, we also integrate denoising training into Deformable DETR with 10 denoising groups as DN-Deformable-DETR. We follow the same setting as Deformable DETR, but specify its query into 4D boxes as in DAB-DETR to better use denoising training.

When comparing in the standard 50 epoch setting, to eliminate any misleading information that the performance improvement of DN-Deformable-DETR may result from the explicit query formulation of anchor boxes, we also implement a strong baseline DAB-Defromable-DETR for comparison. It formulates Deformable DETR query as anchor box without using denoising training, while all the other settings are the same. Note that we strictly follow Deformable DETR to use multi-scale (4 scale) features without FPN. Dynamic DETR [4] add FPN and more scales (5 scales) which can further boost the performance but our performance still exceed theirs.

5.2. Denoising Training Improves Performance

To show the absolute performance improvement compared with DAB-DETR and other single-scale DETR models, we conduct a series of experiments on different backbones under the basic single-scale settings. The results are summarized in Table 1.

The results show that we achieve the best results among single-scale models with all four commonly used backbones. For example, compared with our baseline DAB-DETR under exactly the same setting, we achieve **+1.9** AP absolute improvement with ResNet-50. The table also shows that denoising training adds negligible parameters and computation.

5.3. $1\times$ Setting

With denoising training, the detection task can be accelerated by a large margin. As shown in Table 2, we compare our method with both a traditional detector [15] and some DETR-like models [1, 4, 20]. Note that Dynamic DETR [4] adopts dynamic encoder, for a fair comparison, we also compare with its version without dynamic encoder.

Under the same setting with the DC5-R50 backbone, DN-DETR can outperform DAB-DETR by **+3.7** AP within 12 epochs. Compared with other models, DN-Deformable-DETR achieves the best results in the 12 epoch setting. It is worth noting that our DN-Deformable-DETR achieve **44.1** AP within 12 epochs with the ResNet-101 backbone, which surpasses Faster R-CNN ResNet-101 trained for 108 epochs ($9\times$ faster).

5.4. Compared with State-of-Art Detectors

We also conduct experiments to compare our method with multi-scale models. The results is summarized in Table 3. Our proposed DN-Deformable-DETR achieves the best result **48.6** AP with the ResNet-50 backbone. To eliminate the performance improvement from formulating the queries of deformable DETR as anchor boxes, we further use a strong baseline DAB-Deformable-DETR without denoising training. The results show that we can still yield 1.7 AP absolute improvement. The performance improvement of DN-Deformable-DETR also indicates that denoising training can be integrated into other DETR-like models and improve their performance. Though it is not a fair comparison with Dynamic DETR as it includes dynamic encoder and more scales (5 scales) with FPN, we still yield $+1.4$ AP improvement.

We also show the convergence curve in both single-scale and multi-scale setting in Fig. 5, where we drop learning rate by 0.1 in multiple epochs in Fig. 5(b). The detailed training acceleration analysis and training efficiency is shown is Appendix 7.1 and 7.2.

5.5. Ablation Study

We conduct a series of ablation study with the ResNet-50 backbone trained for 50 epochs to verify the effectiveness each component and report the results in Table 4 and Table 5. The results in Table 4 show that each component in denoising training contributes to the performance improve-

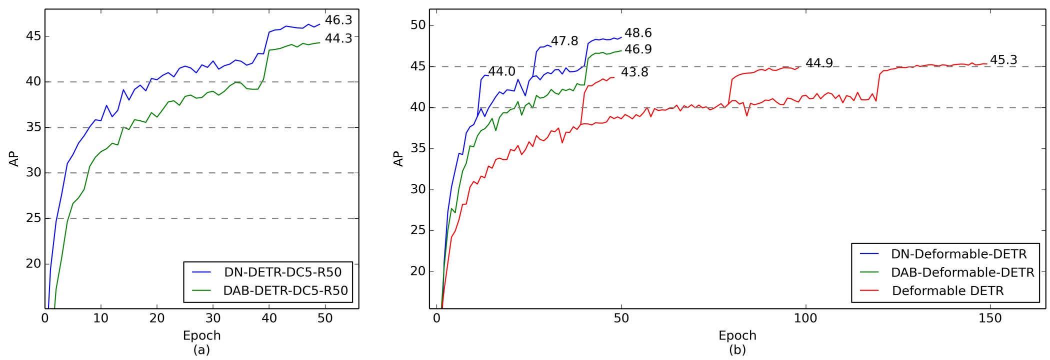

Figure 5. (a) Convergence curves of DAB-DETR and DN-DETR with ResNet-DC5-50. Before learning rate drop, DN-DETR achieves 40 AP in 20 epochs, while DAB-DETR needs 40 epochs. (b) Convergence curves of multi-scale models with ResNet-50. With learning rate drop, DN-Deformable-DETR achieves 47.8 AP in 30 epochs, which is 0.9 AP higher than the converged DAB-Deformable-DETR.

Box Denoising	Label Denoising	Attention Mask	AP
✓	✓	✓	43.4
✓		✓	43.0
		✓	42.2
✓	✓		24.0

Table 4. Ablation results for DN-DETR. All models are trained with the ResNet-50 backbone using 1 denoising group under the same default settings.

	No Group	1 Group	5 Groups
R50	42.2	43.4	44.1
R50-DC5	44.5	45.6	46.3
R101	43.5	45.0	45.2
R101-DC5	45.8	46.5	47.3

Table 5. Ablation results for DN-DETR using different numbers of denoising groups. All models are trained with the ResNet-50 backbone under the same default setting.

ment. Notably, without attention mask to prevent information leakage, the performance degenerates significantly.

We also analyze the influence of the number of denoising groups in our model, as shown in Table 5. The results indicate that adding more denoising groups improves the performance, but the performance improvement becomes marginal as the number of denoising group increases. Therefore, in our experiment, our default setting uses 5 denoising groups, but more denoising groups can further boost performance as well as faster convergence.

In Fig. 6, We explore the influence of noise scale. We run 20 epochs with batch size 64 and ResNet-50 backbone without learning rate drop. The results show that both center shifting and box scaling improve performance. But when the noise is too large, the performance drops.

6. Conclusion

In this paper, we have analyzed the reason for the slow convergence of DETR training lying in the unstable bipartite matching and proposed a novel denoising training method to address this problem. Based on this analysis, we proposed DN-DETR by integrating denoising training into DAB-DETR to test its effectiveness. DN-DETR specifies the decoder embedding as label embedding and introduces denoising training for both boxes and labels. We also added denoisoing training to Deformable DETR to show its generality. The results show that denoising training significantly accelerates convergence and improves performance, leading to the best results in the 1x (12 epochs) setting with both ResNet-50 and ResNet-101 as backbone. This study shows that denoising training can be easily integrated into DETR-like models as a general training method with only a small training cost overhead and bring in a remarkable improvement in terms of both training convergence and detection performance.

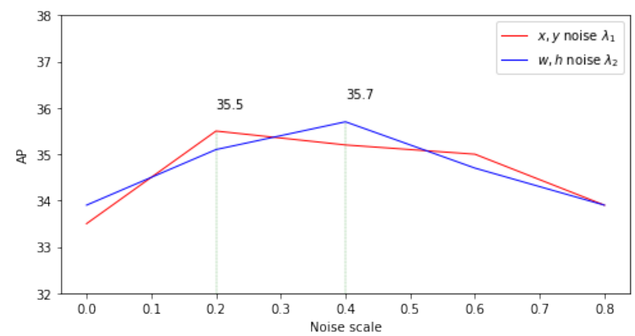

Figure 6. DN-DETR in different noise scales. We fix one noise scale to 0.4 and change the other. Noise scale is defined in 4.3

Limitations and Future Work: In this work, the added noises are simply sampled from uniform distribution. We have not explored more complex noising schemes and leave these for future work. Reconstructing noised data achieves great success in un-supervised learning. This work is an initial step to apply it into object detection. In the future, we will explore how to pretrain detectors on weakly labeled data with unsupervised learning techniques or explore other unsupervised learning methods such as contrastive learning.

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[2] Ting Chen, Saurabh Saxena, Lala Li, David J. Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection, 2021.

[3] Xiyang Dai, Yinpeng Chen, Jianwei Yang, Pengchuan Zhang, Lu Yuan, and Lei Zhang. Dynamic detr: End-to-end object detection with dynamic attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2988–2997, October 2021.

[4] Xiyang Dai, Yinpeng Chen, Jianwei Yang, Pengchuan Zhang, Lu Yuan, and Lei Zhang. Dynamic detr: End-to-end object detection with dynamic attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2988–2997, 2021.

[5] Enrico Maria Fenoaltea, Izat B Baybusinov, Jianyang Zhao, Lei Zhou, and Yi-Cheng Zhang. The stable marriage problem: An interdisciplinary review from the physicist's perspective. *Physics Reports*, 2021.

[6] Peng Gao, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fast convergence of detr with spatially modulated co-attention. *arXiv preprint arXiv:2101.07448*, 2021.

[7] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation, 2014.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[9] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018.

[10] 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.

[11] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. DAB-DETR: Dynamic anchor boxes are better queries for DETR. In *International Conference on Learning Representations*, 2022.

[12] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. *arXiv preprint arXiv:2108.06152*, 2021.

[13] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger, 2016.

[14] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018.

[15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017.

[16] Zhiqing Sun, Shengcao Cao, Yiming Yang, and Kris Kitani. Rethinking transformer-based set prediction for object detection. *arXiv preprint arXiv:2011.10881*, 2020.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

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

[19] Zhuyu Yao, Jiangbo Ai, Boxun Li, and Chi Zhang. Efficient detr: Improving end-to-end object detector with dense prior. *arXiv preprint arXiv:2104.01318*, 2021.

[20] 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: The Ninth International Conference on Learning Representations*, 2021.