StageInteractor: Query-based Object Detector with Cross-stage Interaction

Yao Teng[1] Haisong Liu[1] Sheng Guo[3] Limin Wang[1,2,✉]
[1]State Key Laboratory for Novel Software Technology, Nanjing University, China
[2]Shanghai AI Lab, China [3]MYbank, Ant Group, China

Abstract

Previous object detectors make predictions based on dense grid points or numerous preset anchors. Most of these detectors are trained with one-to-many label assignment strategies. On the contrary, recent query-based object detectors are based a sparse set of learnable queries refined by a series of decoder layers. The one-to-one label assignment is independently applied on each layer for deep supervision during training. Despite the great success of query-based object detection, however, this vanilla one-to-one label assignment strategy requires the detectors to have strong fine-grained discrimination and modeling capacity. In this paper, we propose a new query-based object detector with cross-stage interaction, coined as StageInteractor. During the forward pass, we come up with an efficient way to improve this modeling ability by reusing dynamic operators with lightweight adapters. As for the label assignment, a cross-stage label assigner is designed to improve the one-to-one label assignment. With this assigner, the training target class labels are gathered across stages and then reallocated to proper predictions at each decoder layer. On MS COCO benchmark, our model improves the baseline counterpart by 2.2 AP, and achieves a 44.8 AP with ResNet-50 as backbone, 100 queries and 12 training epochs. With longer training time and 300 queries, StageInteractor achieves 51.3 AP and 52.7 AP with ResNeXt-101-DCN and Swin-S, respectively. The code and models are made available at https://github.com/MCG-NJU/StageInteractor.

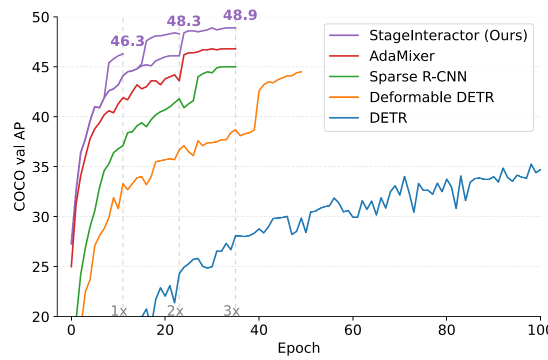


Figure 1: Convergence curves of our model and other query-based object detectors [4, 68, 46, 15] with ResNet-50 [21] on MS COCO [31] minival set.

1. Introduction

Object detection is a fundamental task in computer vision and acts as a cornerstone for many downstream tasks [48, 9]. It aims to localize and categorize all object instances in an image. Over the past few decades, dense spatial prior has been widely applied in various detectors. These detectors make predictions based on either a large quantity of pre-defined anchors covering the whole image [18, 41, 3, 33, 30, 40] or dense grid points in the feature map of this image [26, 13, 65, 49, 39, 57]. To deliver supervision signals to these detectors, most works employ the *one-to-many label assignment* strategy [62, 66, 25, 17, 16, 59] (*i.e.*, the classification label and localization target of one ground-truth object could be assigned to multiple object predictions). Although this paradigm is widely used in object detection, it suffers from redundant and near-duplicate predictions due to such label assignment [45], and thus relies heavily on specific post-processing algorithms for duplicate removal [2, 23, 22].

Recently, DETR [4] and its variants [68, 46, 15, 7, 32, 14, 37, 8] open a new area of object detection. These query-based object detectors get rid of the dense prior and view object detection as a set prediction problem. Specifically, they use a sparse set of *queries* (*i.e.*, learnable embeddings) to progressively capture the characteristics and location of objects with the help of a series of *decoder layers*. In each layer, image features are sampled and fused into the input queries via attention-like operation [51, 68] or dynamic mixing [46, 15]. Then, the transformed queries are decoded into object predictions and also serve as the inputs of next layer. As for the training of this paradigm, a kind of *one-to-*

✉: Corresponding author (lmwang@nju.edu.cn).

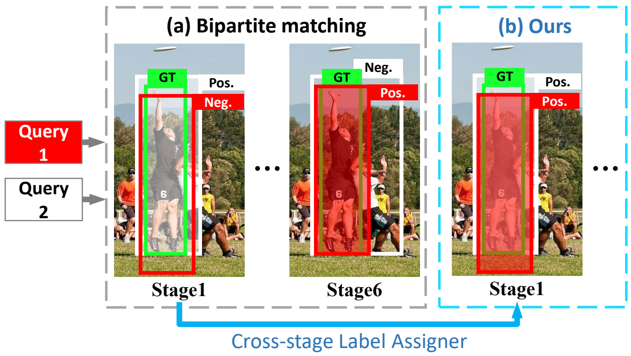


Figure 2: The results of label assignment at various stages. The green box denotes the ground-truth object `Person`. The red and white boxes denote object prediction derived from two different queries. `Pos` and `Neg` denote the positive sample and the negative sample, respectively. (a) The white box is assigned with the ground-truth object `Person` by bipartite matching at the first stage, while the red box is not. But the opposite is true for the sixth stage. (b) With our cross-stage label assigner, the red box in the first stage can be assigned with the ground-truth `Person`.

one label assignment (*i.e.*, each ground-truth object has to be assigned with only one prediction), termed as *bipartite matching*, is independently adopted on each decoder layer for deep supervision [61, 1, 27]. For inference, only the high-confidence outputs from the last layer are taken for evaluation.

However, this one-to-one label assignment requires the detectors to have strong fine-grained discrimination and modeling capacity. On one hand, this strict bipartite matching would ask the detector to capture details to distinguish the predictions. For example, as shown in Fig. 2a, although the predicted boxes of a query (colored in red) cover most of the ground-truth object (`Person`) at each decoder layer, this object is assigned to the boxes of *different* queries at *different* stages (`Stage1`, white box; `Stage6`, red box). In other words, only one of the predicted boxes (red or white) can become the positive sample at each stage. To distinguish these boxes, the detector needs strong fine-grained discriminability to extract high-quality features from the image. Unfortunately, this goal is hard to be fully accomplished. As a result, we are motivated to directly modify the supervision of each decoder layer, *i.e.*, introducing additional supervision from *other stages* to assist the training of this layer, shown in Fig. 2b. In addition, the large modeling capacity is vital to the fine-grained discrimination. To *efficiently* improve this modeling ability, we resort to adding lightweight modules and reusing heavy dynamic operators in decoder layers.

Based on the above analysis, in this paper, we present a new paradigm, *i.e.*, query-based object detector with cross-stage interaction, coined as StageInteractor. This interaction of our method lies in two aspects: cross-stage label assignment and cross-stage dynamic filter reuse. Specifically, during the label assignment, a *cross-stage label assigner* is applied subsequent to the vanilla bipartite matching in each decoder layer. This assigner collects the results of bipartite matching across stages, and then reallocate proper training target labels for each object prediction according to the query index and a score. As for the forward pass, in each decoder layer, we *reuse* the heavy dynamic operators in preceding stages and add lightweight modules to increase the modeling capacity at a relatively low cost.

Experiments show that our model with ResNet-50 [21] as backbone can achieve 44.8 AP on MS COCO validation set under the simple setting of 100 queries, with 27.5 AP$_s$, 48.0 AP$_m$ and 61.3 AP$_l$ on small, medium and large object detection, respectively. Equipped with 3$\times$ training time, 300 queries and more data augmentation in line with other query-based detectors, our model can achieve 49.9 AP, 51.3 AP, and 52.7 AP with ResNet-101, ResNeXt-101-DCN [55, 67], and Swin-S [34] as backbones, under the setting of single scale and single model testing. Our model significantly outperforms the previous methods, as shown in Fig. 1, and it has become a new state-of-the-art query-based object detector.

2. Related Work

DETR [4] is an end-to-end query-based object detector without hand-crafted designs such as preset anchors and non-maximum suppression (NMS), but it suffers from many problems, such as slow training convergence and unstable label assignment. To handle these challenges, a large quantity of works have been proposed. In this part, we divide these methods into two categories: modification of architectures and improvement of training procedures.

Architecture. The vanilla DETR took a transformer encoder-decoder architecture [51] as the detection head, and used a set of object queries, content vectors, to encode the priors of object detection datasets. In Deformable DETR [68], the attention operator [51] was replaced with multi-scale deformable attention module, and iterative bounding box refinement and a two-stage design which enables the detector to adaptively generate queries with a dense prior were also introduced. In Sparse R-CNN [46], object query is decoupled into a content vector and an explicit bounding box, and image features are progressively aggregated into content vectors by ROIAlign [20] with these boxes and dynamic convolution. Moreover, Sparse R-CNN is only a decoder architecture without any transformer encoder. There were also many works like Conditional DETR [36], DAB-DETR [32], SMCA [14] and REGO [8] studying how to introduce spatial priors for proper features to accelerate convergence. Adamixer [15] improved Sparse R-CNN via deformable operators and spa-

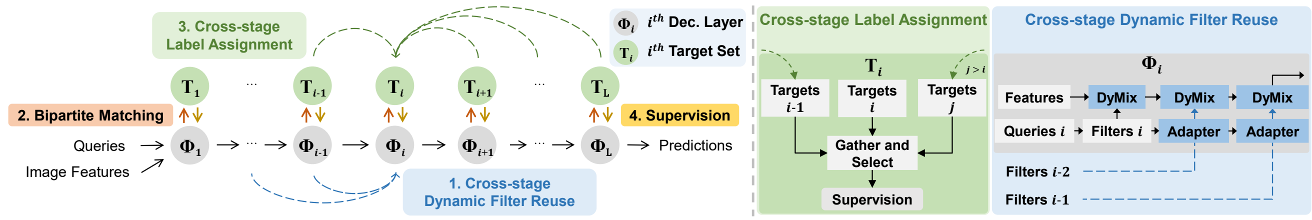


Figure 3: **Overview.** The cross-stage interaction incorporates two parts: cross-stage label assignment and cross-stage dynamic filter reuse. During the forward propagation, dynamic filters in each stage of decode layer are reused in the subsequent stages, *i.e.*, we stack them with specific lightweight adapters to increase the depth of each decoder layer. As for the label assignment, our cross-stage label assigner gathers the results of bipartite matching across stages, and then selects proper target labels as supervision.

tial dynamic convolution to increase the adaptability. There were also works like DDQ [63] which combines the dense and sparse queries to guarantee high recall. In this paper, we improve query-based object detector from a new perspective of the scalability [64, 44, 52] of the decoder, *i.e.*, we reuse dynamic operators among decoder layers to capture more diverse and complex representations.

Training Procedure. In vanilla DETR, a set prediction loss [4] is adopted for training. Recently, many papers have analyzed how to accelerate the convergence of DETR via improved training procedures. To verify whether the instability of Hungarian loss slow down the convergence, Sun *et al.* [47] utilized a matching distillation with a pre-trained DETR providing label assignment to train another model, and they found this instability only influenced the convergence in the early few epochs. DN-DETR [28] presented a denoising training strategy where a set of additional noised ground-truth objects were passed through the decoder to reconstruct the corresponding raw objects. DINO-DETR [60] improved DN-DETR via contrastive denoising training for hard negative samples. Group DETR [6] introduced multiple groups of object queries for the global one-to-many label assignment during training, but maintained one-to-one label assignment in each group, and thus Group DETR could achieve the ability of duplicate removal with one group of queries. Hybrid Matching [24] was also proposed to combine one-to-one and one-to-many label assignment into one query-based object detector with a large number of queries, and it had three types of implementations: hybrid branch scheme (one-to-many matching for one group of queries, one-to-one matching for another), hybrid epoch scheme (one-to-many matching in early epochs, one-to-one matching in late epochs) and hybrid layer scheme (one-to-many matching in early layers, one-to-one in late layers). Different from the previous methods, in this paper, we focus on the calibration of label assignment without adding additional queries. We collect training target labels across stages by a cross-stage label assigner, and then select proper targets to act as the supervision of each object prediction.

3. Proposed Approach

In this paper, we focus on the cross-stage interaction in query-based object detectors because it can well mitigate the misalignment between the decoders and supervisions in an object detector. We first revisit the state-of-the-art query-based object detectors, especially AdaMixer [15], and then elaborate on our proposed cross-stage interaction.

3.1. Preliminary on query-based object detectors

Generally, the architecture of query-based object detectors is composed by four parts: object queries, a backbone, a series of encoders and decoders. Distinctively, Adamixer removes the encoders and still maintains the desired performance. As shown in Fig. 4, it consists of object queries, a backbone and a series of decoders.

Object Query. The initial object queries is just a set of learnable embeddings. Recent object detectors [32, 46, 15] decompose them into content vectors and positional vectors. The content vector is a vector $\mathbf{v} \in \mathbb{R}^D$. The positional vector is presented in the format of box coordinates. For example, in AdaMixer, it contains the information about the center point, scale and aspect ratio of an individual box.

Decoder Layer. In a query-based detector, the decoder layers are stacked multiple times to form a cascade structure. Each decoder layer is generally composed of three components: a multi-head self-attention (MHSA), a dynamic interaction module, and feed-forward networks (FFNs). DETR-like models use the multi-head cross-attention for the dynamic interaction, while AdaMixer adopts a feature sampler and a dynamic mixing module, as shown in Fig. 4. The object queries are sequentially passed through these modules. Specifically, the queries are *first* processed by the multi-head self-attention module. *Then*, its outputs, the updated content vectors, together with the positional vectors are fed into the feature sampler. In this sampler, each query is allocated with a unique group of regional multi-scale image features, *i.e.*, ***sampled features***, by using the queries and bilinear interpolation. *Subsequently*,

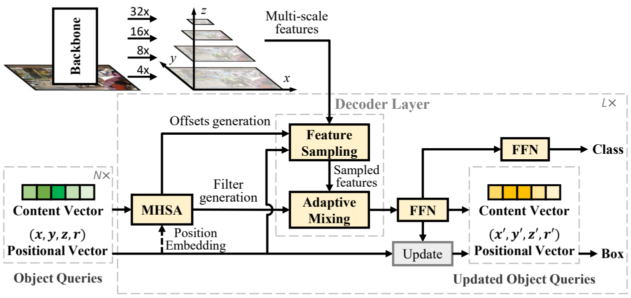


Figure 4: Overview of AdaMixer.

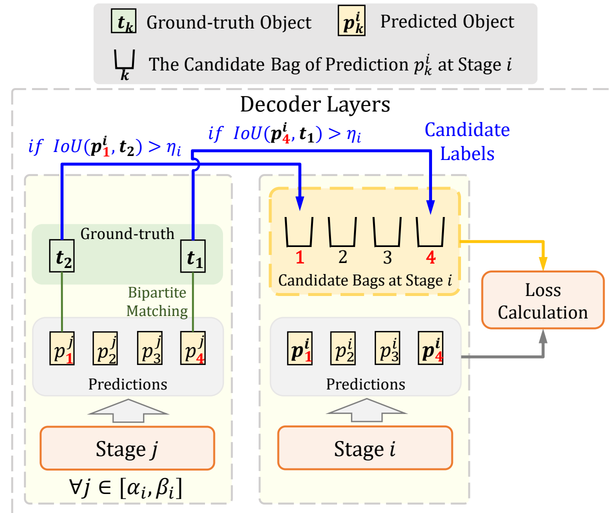
the adaptive mixing is performed on the sampled features with dynamic filters generated from the content vectors. Its outputs are aggregated into the vectors. *Last*, by means of FFNs, the queries updated by these modules can be decoded into object predictions, *i.e*., the relative scaling and offsets to the positional vectors (bounding boxes), and the classification score vectors. These updated queries also serve as inputs of the next stage. Note that any query has and only has one corresponding prediction in every decoder layer. Thus, we simply represent the predictions derived from the same initial query with one index, *i.e*., the ***query index***.

Training and testing. In each decoder layer of a query-based detector, a bipartite matching is directly adopted on the ground-truth objects and the predictions. After some predictions match the ground-truth, these predictions are deemed as positive samples (assigned with foreground classification labels and supervision for localization) while others are the negative (assigned with the background label). Focal loss [30] serves as the classification loss and GIoU loss [42] with ℓ_1 loss acts as the localization loss. During inference, only the outputs with high confidence from the last layer are used for evaluation.

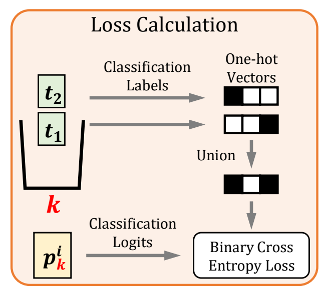
3.2. Cross-stage Label Assignment

To mitigate the training difficulty caused by bipartite matching in query-based object detectors, we propose a new *cross-stage label assigner* to modify the results of the preceding bipartite matching. As depicted in Fig. 3, our assigner first gathers these assignments across the *stages* (*i.e*., decoder layers), and then selects appropriate training target class labels for each prediction.

When gathering training targets, the cross-stage label assigner forces each prediction to only access the targets of predictions sharing the same *query index* (defined in Sec. 3.1) with it. The motivation behind this constraint is that the supervision of a single query may vary across stages, even though its predicted boxes across stages continuously cover the same ground-truth object, shown in Fig. 2a. To alleviate this inconsistency for each query, we leverage its assigned targets provided by bipartite matching from multiple stages. When introducing targets, we use a



(a) Given a stage i, we enumerate other decoder layers whose indexes are denoted as j, and select their assigned targets as the supervision for the stage i according to Eq. (1).



(b) The elements in each candidate set are formed into the final supervision.

Figure 5: The process of our cross-stage label assignment.

score to determine whether a match between a target and a prediction is suitable to be shared across stages.

Algorithm. Thanks to Focal loss [30] with binary cross entropy loss in prevailing query-based object detectors [15, 46, 28, 60], the multi-class classification task can be viewed as the binary classification on each category, and there is no need to consider a separate background category when training. More importantly, *our cross-stage label assignment can be conducted on each category independently*. The specific algorithm is as follows:

Our cross-stage label assignment is performed between two stages with one type of condition. *First*, as shown in Fig. 5a, given a stage i and its predictions $\{p_1^i, ..., p_N^i\}$, we traverse other stages with indexes in the range $[\alpha_i, \beta_i]$. Taking the stage j as an example, we obtain its predictions $\{p_1^j, ..., p_N^j\}$. *Second*, we also create an empty candidate bag for each prediction of the stage i. We perform bipartite matching between the predictions of the stage j and the ground-truth objects. Since this matching is an one-to-one

Method	AP	AP_{50}	AP_{75}
Baseline	43.0	61.5	46.1
Hybrid Layer†	41.8 (-1.2)	60.5 (-1.0)	44.7 (-1.4)
Ours	44.8 (+1.8)	63.0 (+1.5)	48.4 (+2.3)

Table 1: We reproduce (†) the hybrid matching scheme [24], and compare it with our cross-stage label assigner with 100 queries.

label assignment, each ground-truth object corresponds to a prediction with a specific query index at the stage j. *Third*, we align the predictions from these two stages according to the query indexes, and thus each ground-truth object also corresponds to a candidate bag. We then select the training targets from the stage j as the candidate supervisions according to the condition:

$$\vartheta_i(q,t) \geq \eta_i, \tag{1}$$

where $\vartheta_i(q,t)$ denotes a score between the query q and the ground-truth object t at the stage i. Here, the object t needs to be assigned to query q at stage Φ_j by the bipartite matching. η_i denotes a threshold. In practice, we follow the classical setting [41], where the *IoU* as the score and the threshold is set as 0.5. Thus, if the *IoU* is satisfied, the ground-truth objects are added to the corresponding candidate bags. *Finally*, after traversing all the stages in the range $[\alpha_i, \beta_i]$, we use the updated candidate bags to provide supervision for the stage i. As shown in Figure 5b, we convert the labels in each candidate bag into one-hot vectors, and then merge them into one vector. The resulting vector serves as classification supervision through the binary cross entropy loss (Focal loss [30] in practice).

Discussion. Although our label assignment seems to have something in common with existing works like TSP-RCNN [47] and Hybrid Layer Matching [24], the discrepancy between their design and ours cannot be ignored: (1) in TSP-RCNN, first, the idea from Faster-R-CNN [41] is directly borrowed into the set prediction problem (*i.e.*, a ground-truth object is only assigned to the proposal that shares an IoU score greater than 0.5 with it). TSP-RCNN adopts such strategy for both the classification and localization. Differently, we can apply it only for the *classification* task, shown as Tab. 12. Second, TSP-RCNN adopts such strategy with dense proposals for both the classification and localization. Differently, we apply it with *sparse queries*. (2) for Hybrid Layer Matching, ground-truth objects are simply replicated in the first four stages, and then a bipartite matching is conducted. Differently, we do not modify the set of ground-truth objects. We gather results of the bipartite matching across stages, and then only select some targets as supervision. Also, we observe that Hybrid Layer Matching is incompatible with the models given few

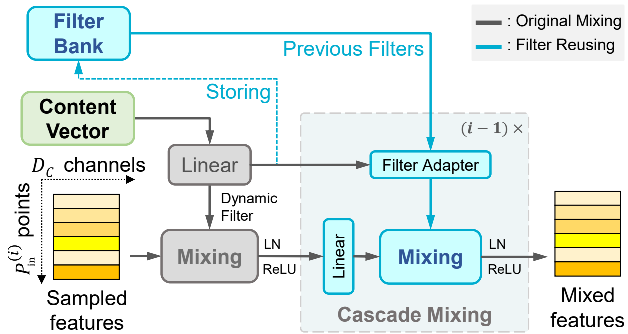


Figure 6: **The dynamic mixing with filter reusing on sampled features**. The content vectors dynamically generate filters through linear layers. These filters are used for mixing on sampled features, and then are stored into a dynamic filter bank for subsequent stages. Previous filters stored in the bank are also reused for mixing.

queries. In Tab. 1, we implement it with the basic setting of our detector with 100 queries. The results show that it is totally infeasible under such a circumstance. In a nutshell, our approach is greatly different from these methods.

3.3. Cross-stage Dynamic Filter Reuse

The model capacity is essential for neural networks to capture complex representations [44, 64]. Since one of the major characteristics shared by query-based detectors is a series of decoder layers, we resort to modifying the decoder to improve this capacity.

A straightforward way of this modification is adding the attention-like operators. The prevailing DETR-like detectors [32, 28, 60] perform deformable cross-attention [68] once along the spatial dimension in each stage, so directly adding this attention is feasible. By contrast, other detectors [46, 15] like AdaMixer perform more than one dynamic mixing at each stage, and they require a huge number of parameters [53] to generate the corresponding dynamic filters. Taking the channel mixing as an example, in each decoder layer, the specific process of generating a filter is as follows:

$$\mathbf{M}_i = \mathbf{W}_0^{(i)} + \sum_{d=1}^{D} \mathbf{W}_d^{(i)} \mathbf{v}_{i,q,d} \in \mathbb{R}^{D_C \times D_C}, \tag{2}$$

where $\mathbf{v}_{i,q,d}$ denotes the d-th element of the content vector of query q at the stage i, $\mathbf{W}_d^{(i)} \in \mathbb{R}^{D_C \times D_C}$ denotes a learnable weight matrix, and the number of such matrices in a stage is D, D_C denotes the channel dimension of the input features, and $\mathbf{M}_i$ serves as the kernel for channel mixing (1×1 convolution) [50]. Only the parameters used to generate a single filter have already been more than $D \times D_C^2$. Thereby, it is impractical to directly stack more decoders given limited resources. Fortunately, there are a large quantity of disposable dynamic filters in these cascade decoder layers, so these operators have the potential to be reused with *lightweight modules* across stages.

As depicted in Fig. 6, we propose a cascade dynamic mixing module with a filter bank as an extension of the original channel mixing. The filters generated in each stage are stored in the filter bank for future use. Given a stage, the *filter adapters* update these stored filters with the ones from the current stage. In each adapter, the process of updating filters is as follows:

$$\mathbf{w}_j^{(1)} = \sigma(\mathbf{W}_j^{(1)}\mathbf{v}_{i,q}) \in \mathbb{R}^{D_C}, \mathbf{w}_j^{(2)} = \sigma(\mathbf{W}_j^{(2)}\mathbf{v}_{i,q}) \in \mathbb{R}^{D_C},$$
$$\mathbf{M}'_{j,i} = \left(\mathbf{w}_j^{(1)} \cdot \mathbf{w}_j^{(2)\top}\right) \odot \mathbf{M}_j + \left(\mathbf{1} - \mathbf{w}_j^{(1)} \cdot \mathbf{w}_j^{(2)\top}\right) \odot \mathbf{M}_i, \tag{3}$$

where $\mathbf{1}$ denotes an all-ones matrix, $\mathbf{W}_j^{(1)}, \mathbf{W}_j^{(2)}$ denote learnable weights, $\sigma(\cdot)$ denotes the sigmoid function, $\odot$ denotes the element-wise product, $\mathbf{M}_j$ denotes a previous filter, $j \in [\gamma_i, i)$, γ_i is a lower bound, $\mathbf{M}'_{j,i}$ is used for subsequent modules, and the number of these updated filters is $\Delta\gamma_i = i - \gamma_i$. We empirically find more filters can lead to more performance gain, so $\Delta\gamma_i = i - 1$ is the best choice. Note that this adapter only costs $2D_C^2$ parameters, which is more lightweight than the process of generating a filter from scratch as Eq. (2). Then, the updated filters are used in the cascade dynamic mixing. In this structure, since each dynamic mixing is followed by a layer normalization with an activation function, we insert a lightweight linear layer between every two dynamic mixing, consistent with [21].

Moreover, we can modify the original spatial mixing with the dynamic filter reusing. Unlike the channel mixing, the original dynamic spatial mixing costs a lot of computational resources on its ouput features due to its expansion [43] on spatial dimension. To tackle this problem, we also adopt the bank to store the filters generated for spatial mixing. Then, in each decoder layer, we update these filters with adapters and concatenate them with the current filter along the output dimension. This new filter is used to perform spatial mixing only once, rather than iteratively. This approach allows us to reduce the parameters of the filter generator of the spatial mixing, *i.e.*, we employ the filters from preceding stages to replace a proportion of the filter from the current layer.

4. Experiments

4.1. Implementation Details

The experiments are performed on the MS COCO [31] object detection dataset, where the train2017 split and the val2017 split are for training and testing. All of our experiments on AdaMixer are based on mmdetection codebase [5]. The experiments on DN-DETR and DINO are based on DETREX codebase [11]. The convolutional neural networks [21, 56] or vision trasnformers [19, 34, 12] can be taken as the backbone network. 8 RTX 2080ti with 11G GPU memory are enough to train our model with ResNet-50 and 100 queries. For larger backbones or more queries, we resort to 8 V100 with 32G. The cross-stage label assignment is applied on each decoder layer. The reuse of spatial dynamic filters do not perform on the first two decoder layers. The weighted sum of these losses are for training, and the loss weights are in line with those of AdaMixer in mmdetection codebase [5] and those of DETRs in DETREX codebase [11]. AdamW [35] is taken as the optimizer.

detector	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
FCOS [49]	38.7	57.4	41.8	22.9	42.5	50.1
Cascade R-CNN [3]	40.4	58.9	44.1	22.8	43.7	54.0
GFocalV2 [29]	41.1	58.8	44.9	23.5	44.9	53.3
BorderDet [38]	41.4	59.4	44.5	23.6	45.1	54.6
Dynamic Head [10]	42.6	60.1	46.4	26.1	46.8	56.0
DETR [4]	20.0	36.2	19.3	6.0	20.5	32.2
Deform-DETR [68]	35.1	53.6	37.7	18.2	38.5	48.7
Sparse R-CNN [46]	37.9	56.0	40.5	20.7	40.0	53.5
AdaMixer [15]	42.7	61.5	45.9	24.7	45.4	59.2
AdaMixer† [15]	45.0	64.2	48.6	27.9	47.8	61.1
StageInteractor	**44.8**	**63.0**	**48.4**	**27.5**	**48.0**	**61.3**
StageInteractor†	**46.9**	**65.2**	**51.1**	**30.0**	**49.7**	**62.3**

Table 2: $1\times$ **training scheme (12 epochs)** performance of various detectors on COCO `minival` set with ResNet-50. 100 object queries is the default setting in our method. † denotes 500 queries.

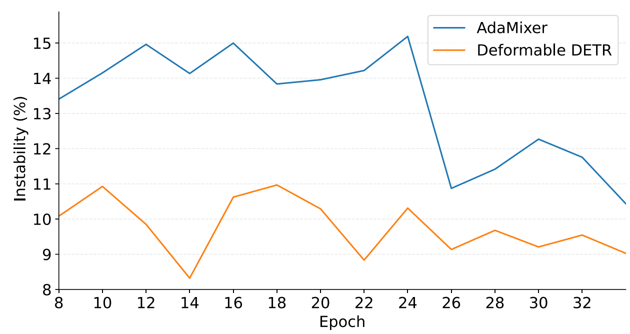


Figure 7: The instability of the assigned labels.

4.2. Comparison to State-of-the-Art Detectors

Our model significantly outperforms the previous methods, shown in Fig. 1, and it has become a new state-of-the-art query-based object detector. As shown in Tab. 2, with $1\times$ training scheme and ResNet-50 [21] as backbone, our detector outperforms various methods on COCO `minival` set [31] with even with 100 queries. Our model with ResNet-50 as backbone can achieve 44.8 AP on MS COCO validation set under the basic setting of 100 object queries, with 27.5 AP_s, 48.0 AP_m and 61.3 AP_l on small, medium and large object detection, respectively. When

Detector	Backbone	Encoder/FPN	Epochs	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
DETR [4]	ResNet-50-DC5	TransformerEnc	500	43.3	63.1	45.9	22.5	47.3	61.1
SMCA [14]	ResNet-50	TransformerEnc	50	43.7	63.6	47.2	24.2	47.0	60.4
Deformable DETR [68]	ResNet-50	DeformTransEnc	50	43.8	62.6	47.7	26.4	47.1	58.0
Anchor DETR [54]	ResNet-50-DC5	DecoupTransEnc	50	44.2	64.7	47.5	24.7	48.2	60.6
Efficient DETR [58]	ResNet-50	DeformTransEnc	**36**	45.1	63.1	49.1	28.3	48.4	59.0
Conditional DETR [36]	ResNet-50-DC5	TransformerEnc	108	45.1	65.4	48.5	25.3	49.0	62.2
Sparse R-CNN [46]	ResNet-50	FPN	**36**	45.0	63.4	48.2	26.9	47.2	59.5
REGO [8]	ResNet-50	DeformTransEnc	50	47.6	66.8	51.6	29.6	50.6	62.3
DAB-D-DETR [32]	ResNet-50	DeformTransEnc	50	46.8	66.0	50.4	29.1	49.8	62.3
DN-DAB-D-DETR [28]	ResNet-50	DeformTransEnc	**12**	43.4	61.9	47.2	24.8	46.8	59.4
DN-DAB-D-DETR [28]	ResNet-50	DeformTransEnc	50	48.6	67.4	52.7	31.0	**52.0**	63.7
AdaMixer [15]	ResNet-50	-	**12**	44.1	63.1	47.8	29.5	47.0	58.8
AdaMixer [15]	ResNet-50	-	**24**	46.7	65.9	50.5	29.7	49.7	61.5
AdaMixer [15]	ResNet-50	-	**36**	47.0	66.0	51.1	30.1	50.2	61.8
StageInteractor	ResNet-50	-	**12**	46.3	64.3	50.6	29.8	49.6	60.8
StageInteractor	ResNet-50	-	**24**	48.3	66.6	52.9	31.7	51.4	63.3
StageInteractor	ResNet-50	-	**36**	**48.9**	**67.4**	**53.4**	**31.7**	51.8	**64.3**
StageInteractor*	ResNet-50	-	**36**	**50.8**	**66.8**	**55.9**	**34.0**	**54.6**	**66.2**
DETR [4]	ResNet-101-DC5	TransformerEnc	500	44.9	64.7	47.7	23.7	49.5	62.3
SMCA [14]	ResNet-101	TransformerEnc	50	44.4	65.2	48.0	24.3	48.5	61.0
Efficient DETR [58]	ResNet-101	DeformTransEnc	**36**	45.7	64.1	49.5	28.2	49.1	60.2
Conditional DETR [36]	ResNet-101-DC5	TransformerEnc	108	45.9	66.8	49.5	27.2	50.3	63.3
Sparse R-CNN [46]	ResNet-101	FPN	**36**	46.4	64.6	49.5	28.3	48.3	61.6
REGO [8]	ResNet-101	DeformTransEnc	50	48.5	67.0	52.4	29.5	52.0	64.4
AdaMixer [15]	ResNet-101	-	**36**	48.0	67.0	52.4	30.0	51.2	63.7
StageInteractor	ResNet-101	-	**36**	**49.9**	**68.6**	**54.6**	**33.0**	**53.6**	**65.4**
REGO [8]	ResNeXt-101	DeformTransEnc	50	49.1	67.5	53.1	30.0	52.6	65.0
AdaMixer [15]	ResNeXt-101-DCN	-	**36**	49.5	68.9	53.9	31.3	52.3	66.3
StageInteractor	ResNeXt-101-DCN	-	**36**	**51.3**	**70.2**	**56.0**	**33.2**	**54.6**	**66.9**
AdaMixer [15]	Swin-S	-	**36**	51.3	71.2	55.7	34.2	54.6	67.3
StageInteractor	Swin-S	-	**36**	**52.7**	**71.7**	**57.7**	**36.1**	**56.2**	**67.7**

Table 3: The performance of various query-based detectors on MS COCO `minival` set with longer training scheme and single scale testing. The number of queries defaults to 300 in our method. * denotes 900 queries with more sampling points.

equppied with 500 queries, our detector performs better, and it can achieve 46.9 AP. As depicted in Tab. 3, equipped with 3× training time, 300 queries and more data augmentation in line with other query-based object detectors, our model can achieve 48.9 AP, 49.9 AP, 51.3 AP, and 52.7 AP with ResNet-50, ResNet-101, ResNeXt-101-DCN [55, 67], and Swin-S [34] as backbones, under the setting of single scale and single model testing. Moreover, if we extend the quantity of queries into 900 and use tricks (*i.e.*, adding more sampling points at each stage), our model can achieve 50.8 AP with ResNet-50.

More importantly, our designs can be applied on DETR-like detectors such as DN-DETR [28] and DINO [60]. Unlike the cross-stage label assignment, our cross-stage filter reuse is tailored for AdaMixer [15]. To achieve larger model capacity, we opt to simply double the cross-attention because this attention-like operation is more lightweight than the dynamic mixing. In Tab. 4, our designs yield more than +0.5AP for all detectors. To verify the effectiveness of our cross-stage label assignment on the larger backbones, we use DINO with Swin-B [34] for experiments. As shown in Tab. 5, our cross-stage label assignment can still yield +0.4AP on DINO with Swin-B as backbone with 12 epochs.

To find the reason for the lower performance gain on DETRs than that on AdaMixer, we present the instability of assigned labels (*i.e.*, the probability of a ground-truth object transferring from one query to another across stages) as curves in Fig. 7. This figure shows that Deformable-DETR is more stable, and we attribute this to its powerful transformer encoder which provides high-quality features for the

Detector	Epoch	AP	AP_{50}	AP_{75}
DINO [60]	12	49.2	66.9	53.7
DINO + CSLA	12	49.7	67.0	54.1
DINO + CSLA + DCA	12	**50.0**	**67.4**	**54.9**
DINO [60]	24	50.6	68.5	55.5
DINO + CSLA	24	**51.0**	68.7	55.6
DINO + CSLA + DCA	24	**51.3**	**69.2**	**56.1**
Deform-DETR [68]	50	46.1	64.9	49.9
Deform-DETR + CSLA	50	**46.8**	**65.0**	**50.9**
DN-DETR [28]	50	44.7	**65.3**	47.6
DN-DETR + CSLA	50	**45.5**	**65.3**	**49.0**
H-DETR [24]	12	48.6	**66.3**	53.2
H-DETR + CSLA	12	**49.2**	**66.3**	**53.7**

Table 4: The performance of DINO [60] on MS COCO `minival` set with ResNet-50. `CSLA`: cross-stage label assignment. `DCA`: dual cross-attention.

Detector	Backbone	AP	AP_{50}	AP_{75}
DINO [60]	Swin-B	55.8	74.4	60.7
DINO [60] + CSLA	Swin-B	**56.2**	74.7	61.3

Table 5: Cross-stage label assignment on DINO [60] with Swin-B [34] as backbone.

CSLA	Reuse	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
		42.6	61.4	45.7	24.4	45.7	58.2
	✓	43.0	61.5	46.1	25.0	45.7	59.1
✓		44.1	62.3	47.6	25.5	47.5	60.3
✓	✓	**44.8**	**63.0**	**48.4**	**27.5**	**48.0**	**61.3**

Table 6: The effectiveness of our proposed modules. `CSLA`: cross-stage label assignment.

decoder.

4.3. Ablation Studies

Because the computational resource is limited, ResNet-50 is employed as our backbone network, the number of queries is set as 100, and $1\times$ training scheme is used for our ablation studies. Due to the simplicity, we adopt AdaMixer as the baseline for ablation studies.

The effectiveness of our proposed modules. In Tab. 6, we conduct experiments to verify the effectiveness of our proposed modules. The results show that our modules lead to +2.2 AP gain, and each of them boosts the performance.

The components of our label assigner. We verify how the components of our cross-stage label assigner influences the performance in Tab. 7. In the first two lines, the results show that directly adding more supervisions based on the query index only brings marginal gains. According to the last two lines, we find that while using scores like IoU can boost the performance, using both IoU and the cross-stage labels leads to better results.

The number of reused spatial filters. We explore the effectiveness of reusing spatial filters and the results are in Tab. 8. We find a certain number of reused filters can bring a slight performance gain. This may result from the easier optimization brought by the fewer model parameters.

Inference speed and training memory use. As depicted in Tab. 9, with one TITAN XP and one batch size, the speed of our model is 11.6 img/s while that of the baseline is 13.5 img/s, *i.e.*, only $\mathbf{1.16}\times$ slower. When we set the batch size as 2 for training, the additional operation only costs about 0.3G GPU memory (about **5**%).

The number of additional channel mixing. We conduct ablation studies on how many previous filters are required for the channel mixing, and report performance in Tab. 10. We do experiments on the $\max\{\Delta\gamma_i\}$. This means there are at most $\max\{\Delta\gamma_i\}$ filters reused for the i-th stage. We find more filters can bring more performance gain. The results also show the scalability of the decoder layers in this framework.

Selection of filters on mixing. We explore whether the previous filters with adapters are suitable for our new channel mixing. As shown in Tab. 11, we find that using the adapters to fuse previous filters with the current ones is most suitable, which ensures both the adaptability of each stage and the diversity [64] of filters. More importantly, the existence of the additional dynamic mixing is necessary.

Modifying the localization supervision. In our cross-stage label assigner, only classification labels are used for gathering, while the supervision for localization is unchanged. Thus, we explore its influence by consistently updating the supervision of classification and localization, *i.e.* selecting the ground-truth boxes that have the greatest IoU with predicted boxes across stages. The results are shown in Tab. 12, and we find that the performance under the two settings is very close, so we do not modify the localization part for simplicity.

Application scope of the cross-stage label assigner. For the i-th stage, the application scope of cross-stage label assigner is $[\alpha_i, \beta_i]$. In this part, we explore the appropriate values of α_i and β_i. As shown in Tab. 13, we find that the best setting is $[i-1, L]$. Yet if too many previous ground-truth labels are included, like $\alpha_i = 1$, this hinders the last decoder layer to learn how to remove duplicates. Moreover, we conduct experiments to verify whether the cross-stage label assigner needs to be applied on all stages, because existing works [4, 24] demonstrate that only early stages can be applied with one-to-many label assignment. On the contrary, as shown in the last line of Tab. 13, we find that our

Cross	Score	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
		43.0	61.5	46.1	25.0	45.7	59.1
✓		43.1	61.9	46.5	25.3	45.8	59.7
	✓	44.0	62.1	47.6	26.0	47.6	59.3
✓	✓	**44.8**	**63.0**	**48.4**	**27.5**	**48.0**	**61.3**

Table 7: The effectiveness of the components of the cross-stage label assigner.

N_S	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
0	44.5	62.5	48.1	27.4	48.0	60.6
1	**44.8**	**63.0**	**48.4**	**27.5**	**48.0**	**61.3**
3	44.4	62.4	48.0	26.2	47.4	60.6

Table 8: The number of reused spatial filters.

Method	AP	Speed (img/s)	Memory (GB)
Baseline	42.6	13.5	6.0
Ours	44.8	11.6	6.3

Table 9: The inference speed and training GPU memory use.

$\max\{\Delta\gamma_i\}$	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
0	44.1	62.3	47.6	25.5	47.5	60.3
1	44.2	62.3	47.6	25.8	47.5	60.6
2	44.2	62.3	47.9	26.1	48.0	60.4
3	44.6	62.8	48.3	26.7	47.5	60.7
4	**44.8**	**63.0**	**48.4**	**27.5**	**48.0**	**61.3**

Table 10: The number of additional channel mixing.

Filter	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
	43.5	61.7	46.8	25.3	47.0	59.0
Pre	44.2	61.9	47.6	26.1	47.6	60.0
Cur	44.5	62.7	47.9	26.5	47.7	60.5
Adp	**44.8**	**63.0**	**48.4**	**27.5**	**48.0**	**61.3**

Table 11: Usage of filters. `Pre`, `Cur`: directly using previous and current filters. `Adp`: using adapters.

cross-stage label assigner can also be applied on the last two stages and this brings performance gain. We speculate that this module helps remove some inappropriate supervision caused by vanilla bipartite matching which has a shortage of constraining IoU, and this module provides some proper supervision from the previous stage.

The threshold of IoU. As shown in Tab. 14, we find that using a threshold of 0.5 is the best choice to select appropriate labels.

Loc.	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
	44.8	62.8	**48.4**	26.7	**48.2**	60.7
✓	**44.8**	**63.0**	**48.4**	**27.5**	48.0	**61.3**

Table 12: Adding localization supervision in cross-stage label assigner.

Range of i	α_i	β_i	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
$[1, L]$	i	i	44.0	62.1	47.6	26.0	47.6	59.3
$[1, L]$	$i-1$	i	44.1	62.2	47.7	25.9	47.5	59.8
$[1, L]$	$i-1$	$i+1$	44.1	62.4	48.0	26.5	47.3	60.1
$[1, L]$	$i-1$	L	**44.8**	**63.0**	**48.4**	**27.5**	**48.0**	**61.3**
$[1, L]$	1	L	43.3	60.7	46.8	25.7	47.2	58.6
$[1, L-2]$	$i-1$	L	44.3	62.4	47.7	26.1	47.6	60.4

Table 13: The application scope of the cross-stage label assigner for each stage.

IoU thres	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l
0.3	43.8	62.4	47.2	26.3	46.8	60.3
0.7	44.4	61.1	48.4	26.0	47.5	61.5
0.5	**44.8**	**63.0**	**48.4**	27.5	48.0	61.3

Table 14: The threshold of IoU in cross-stage label assignment.

5. Conclusion

In this paper, we have presented a new fast-converging query-based object detector with cross-stage interaction, termed as StageInteractor, to alleviate inconsistency between the one-to-one label assignment and improve the modeling capacity. Our proposed cross-stage label assigner gathers assigned labels across stages, and then reassigns proper labels to predictions in each stage. We also accumulate dynamic filter across stages and reuse them with lightweight modules to increase the modeling capacity without introducing too many parameters. With these two unique designs, StageInteractor significantly outperforms the previous methods, and our proposed label assignment can boost the performance of various query-based object detectors.

Acknowledgements. This work is supported by National Key R&D Program of China (No. 2022ZD0160900), National Natural Science Foundation of China (No. 62076119, No. 61921006), Fundamental Research Funds for the Central Universities (No. 020214380091, No. 020214380099), and Collaborative Innovation Center of Novel Software Technology and Industrialization. The authors would like to thank Ziteng Gao for helpful discussions.

References

[1] Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. In *Proceedings of the AAAI conference on artificial intelligence*, pages 3159–3166, 2019. 2

[2] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms–improving object detection with one line of code. In *Proceedings of the IEEE international conference on computer vision*, pages 5561–5569, 2017. 1

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: high quality object detection and instance segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 43(5):1483–1498, 2019. 1, 6

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European conference on computer vision*, pages 213–229, 2020. 1, 2, 3, 6, 7, 8

[5] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 6

[6] Qiang Chen, Xiaokang Chen, Gang Zeng, and Jingdong Wang. Group detr: Fast training convergence with decoupled one-to-many label assignment. *arXiv preprint arXiv:2207.13085*, 2022. 3

[7] Xiaokang Chen, Fangyun Wei, Gang Zeng, and Jingdong Wang. Conditional detr v2: Efficient detection transformer with box queries. *arXiv preprint arXiv:2207.08914*, 2022. 1

[8] Zhe Chen, Jing Zhang, and Dacheng Tao. Recurrent glimpse-based decoder for detection with transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5260–5269, 2022. 1, 2, 7

[9] Yuren Cong, Wentong Liao, Hanno Ackermann, Bodo Rosenhahn, and Michael Ying Yang. Spatial-temporal transformer for dynamic scene graph generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16372–16382, 2021. 1

[10] Xiyang Dai, Yinpeng Chen, Bin Xiao, Dongdong Chen, Mengchen Liu, Lu Yuan, and Lei Zhang. Dynamic head: Unifying object detection heads with attentions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7373–7382, 2021. 6

[11] detrex contributors. detrex: An research platform for transformer-based object detection algorithms, 2022. 6

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 6

[13] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6569–6578, 2019. 1

[14] Peng Gao, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fast convergence of detr with spatially modulated co-attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3621–3630, 2021. 1, 2, 7

[15] Ziteng Gao, Limin Wang, Bing Han, and Sheng Guo. Adamixer: A fast-converging query-based object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5364–5373, 2022. 1, 2, 3, 4, 5, 6, 7

[16] Ziteng Gao, Limin Wang, and Gangshan Wu. Mutual supervision for dense object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3641–3650, 2021. 1

[17] Zheng Ge, Songtao Liu, Zeming Li, Osamu Yoshie, and Jian Sun. Ota: Optimal transport assignment for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 303–312, 2021. 1

[18] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 1

[19] Meng-Hao Guo, Cheng-Ze Lu, Zheng-Ning Liu, Ming-Ming Cheng, and Shi-Min Hu. Visual attention network. *arXiv preprint arXiv:2202.09741*, 2022. 6

[20] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 2, 6

[22] Yihui He, Chenchen Zhu, Jianren Wang, Marios Savvides, and Xiangyu Zhang. Bounding box regression with uncertainty for accurate object detection. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 2888–2897, 2019. 1

[23] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3588–3597, 2018. 1

[24] Ding Jia, Yuhui Yuan, Haodi He, Xiaopei Wu, Haojun Yu, Weihong Lin, Lei Sun, Chao Zhang, and Han Hu. Detrs with hybrid matching. *arXiv preprint arXiv:2207.13080*, 2022. 3, 5, 8

[25] Kang Kim and Hee Seok Lee. Probabilistic anchor assignment with iou prediction for object detection. In *European Conference on Computer Vision*, pages 355–371. Springer, 2020. 1

[26] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European conference on computer vision*, pages 734–750, 2018. 1

[27] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial intelligence and statistics*, pages 562–570. PMLR, 2015. 2

[28] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M Ni, and Lei Zhang. Dn-detr: Accelerate detr training by introducing query denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13619–13627, 2022. 3, 4, 5, 7, 8

[29] Xiang Li, Wenhai Wang, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss v2: Learning reliable localization quality estimation for dense object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11632–11641, 2021. 6

[30] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 1, 4, 5

[31] 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. 1, 6

[32] 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. *arXiv preprint arXiv:2201.12329*, 2022. 1, 2, 3, 5, 7

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

[34] 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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 2, 6, 7, 8

[35] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[36] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3651–3660, 2021. 2, 7

[37] Duy-Kien Nguyen, Jihong Ju, Olaf Booij, Martin R Oswald, and Cees GM Snoek. Boxer: Box-attention for 2d and 3d transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4773–4782, 2022. 1

[38] Han Qiu, Yuchen Ma, Zeming Li, Songtao Liu, and Jian Sun. Borderdet: Border feature for dense object detection. In *European Conference on Computer Vision*, pages 549–564. Springer, 2020. 6

[39] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 1

[40] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. 1

[41] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 1, 5

[42] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019. 4

[43] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 6

[44] Zhiqiang Shen, Zechun Liu, and Eric Xing. Sliced recursive transformer. *arXiv preprint arXiv:2111.05297*, 2021. 3, 5

[45] Peize Sun, Yi Jiang, Enze Xie, Wenqi Shao, Zehuan Yuan, Changhu Wang, and Ping Luo. What makes for end-to-end object detection? In *International Conference on Machine Learning*, pages 9934–9944. PMLR, 2021. 1

[46] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14454–14463, 2021. 1, 2, 3, 4, 5, 6, 7

[47] 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. 3, 5

[48] Yao Teng and Limin Wang. Structured sparse r-cnn for direct scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19437–19446, 2022. 1

[49] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9627–9636, 2019. 1, 6

[50] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34:24261–24272, 2021. 5

[51] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1, 2

[52] Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. Deepnet: Scaling transformers to 1,000 layers. *arXiv preprint arXiv:2203.00555*, 2022. 3

[53] Wen Wang, Jing Zhang, Yang Cao, Yongliang Shen, and Dacheng Tao. Towards data-efficient detection transformers. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IX*, pages 88–105. Springer, 2022. 5

[54] Yingming Wang, Xiangyu Zhang, Tong Yang, and Jian Sun. Anchor detr: Query design for transformer-based detector. In *Proceedings of the AAAI conference on artificial intelligence*, pages 2567–2575, 2022. 7

[55] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 2, 7

[56] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 6

[57] Ze Yang, Shaohui Liu, Han Hu, Liwei Wang, and Stephen Lin. Reppoints: Point set representation for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9657–9666, 2019. 1

[58] 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. 7

[59] Mohsen Zand, Ali Etemad, and Michael Greenspan. Objectbox: From centers to boxes for anchor-free object detection. *arXiv preprint arXiv:2207.06985*, 2022. 1

[60] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022. 3, 4, 5, 7, 8

[61] Linfeng Zhang, Xin Chen, Junbo Zhang, Runpei Dong, and Kaisheng Ma. Contrastive deep supervision. In *European Conference on Computer Vision*, pages 1–19. Springer, 2022. 2

[62] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9759–9768, 2020. 1

[63] Shilong Zhang, Xinjiang Wang, Jiaqi Wang, Jiangmiao Pang, Chengqi Lyu, Wenwei Zhang, Ping Luo, and Kai Chen. Dense distinct query for end-to-end object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7329–7338, 2023. 3

[64] Daquan Zhou, Bingyi Kang, Xiaojie Jin, Linjie Yang, Xiaochen Lian, Zihang Jiang, Qibin Hou, and Jiashi Feng. Deepvit: Towards deeper vision transformer. *arXiv preprint arXiv:2103.11886*, 2021. 3, 5, 8

[65] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 1

[66] Benjin Zhu, Jianfeng Wang, Zhengkai Jiang, Fuhang Zong, Songtao Liu, Zeming Li, and Jian Sun. Autoassign: Differentiable label assignment for dense object detection. *arXiv preprint arXiv:2007.03496*, 2020. 1

[67] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9308–9316, 2019. 2, 7

[68] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 1, 2, 5, 6, 7, 8