DETRs with Hybrid Matching

Ding Jia[1†]   Yuhui Yuan[4†‡]   Haodi He[2†]   Xiaopei Wu[3]

Haojun Yu[1]   Weihong Lin[4]   Lei Sun[4]   Chao Zhang[1]   Han Hu[4]

[1]Peking University   [2]Stanford University   [3]Zhejiang University

[4]Microsoft Research Asia

Abstract

One-to-one set matching is a key design for DETR to establish its end-to-end capability, so that object detection does not require a hand-crafted NMS (non-maximum suppression) to remove duplicate detections. This end-to-end signature is important for the versatility of DETR, and it has been generalized to broader vision tasks. However, we note that there are few queries assigned as positive samples and the one-to-one set matching significantly reduces the training efficacy of positive samples. We propose a simple yet effective method based on a hybrid matching scheme that combines the original one-to-one matching branch with an auxiliary one-to-many matching branch during training. Our hybrid strategy has been shown to significantly improve accuracy. In inference, only the original one-to-one match branch is used, thus maintaining the end-to-end merit and the same inference efficiency of DETR. The method is named $\mathcal{H}$-DETR, and it shows that a wide range of representative DETR methods can be consistently improved across a wide range of visual tasks, including Deformable-DETR, PETRv2, PETR, and TransTrack, among others. Code is available at: https://github.com/HDETR.

1. Introduction

Since the success of pioneering work DEtection TRansformer (DETR) [5] on object detection tasks, DETR-based approaches have achieved significant progress on various fundamental vision recognition tasks such as object detection [50, 58, 82, 89], instance segmentation [13, 14, 25, 78], panoptic segmentation [9, 31, 64, 75, 79], referring expression segmentation [69, 74], video instance segmentation [8, 65, 70], pose estimation [26, 59, 60], multi-object tracking [7, 49, 61], monocular depth estimation [18, 29], text detection & layout analysis [46, 55, 56, 85], line segment detection [71], 3D object detection based on point clouds

or multi-view images [1, 30, 52, 67], visual question answering [22, 47], and so on.

Many follow-up efforts have improved DETR from various aspects, including redesigning more advanced transformer encoder [15, 89] or transformer decoder architectures [4, 16, 50, 81, 89] or query formulations [24, 37, 68, 82]. Different from most of these previous efforts, we focus on the training efficacy issues caused by one-to-one matching, which only assigns one query to each ground truth. For example, Deformable-DETR typically only selects less than 30 queries from a pool of 300 queries to match with the ground truth for each image, as nearly 99% of the COCO images consist of less than 30 bounding boxes annotations, while the remaining more than 270 queries will be assigned as $\varnothing$ and are supervised with only classification loss, thus suffering from very limited localization supervision.

To overcome the drawbacks of one-to-one matching and unleash the benefits of exploring more positive queries, we present a very simple yet effective hybrid matching scheme, which introduces an additional one-to-many matching branch that assigns multiple queries to each positive sample. In inference, we only use the original one-to-one decoder branch supervised with the one-to-one matching loss. We find that this simple approach can substantially improve the training efficacy, especially regarding the fitting of positive queries. Since only the original one-to-one matching branch is used in inference, the merits of the original DETR framework are almost all maintained, for example, avoiding NMS. Our approach also has no additional computation overhead compared to the original version.

We dub the hybrid matching approach as $\mathcal{H}$-DETR, and extensively verify its effectiveness using a variety of vision tasks that adopt DETR methods or the variants, as well as different model sizes ranging from ResNet-50/Swin-T to Swin-L. The visual tasks and the corresponding DETR-based approaches include Deformable-DETR [89] for image object detection, PETRv2 [40] for 3D object detection from multi-view images, PETR [59] for multi-person pose estimation, and TransTrack [61] for multi-object tracking. The $\mathcal{H}$-DETR achieves consistent gains over all of them, as

---

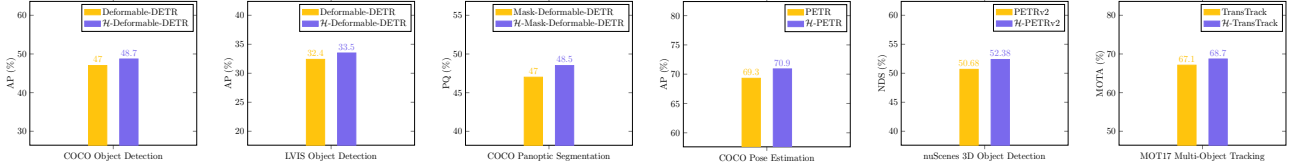[†]Core contribution.

[‡]Corresponding author

Figure 1. Illustrating the improvements of our hybrid matching scheme across five challenging vision tasks including 2D object detection, 2D panoptic segmentation, 2D pose estimation, 3D object detection, and multi-object tracking (from left to right). Our hybrid matching scheme gains +1.7%, +1.1%, +1.5%, +1.6%, +1.7%, and +1.6% over various DETR-based approaches on $6\times$ benchmarks respectively. All the improvements are obtained under the same training epochs and do not require any additional computation cost during evaluation. We choose VoVNetV2 [23]/ResNet50 for PETRv2/all other methods as the backbone following their original settings.

shown in Figure 1. Specifically, our approach can improve the Deformable DETR framework (R50) on COCO object detection by +1.7% mAP (48.7% v.s. 47.0%), the PETR framework (R50) on COCO pose estimation by +1.6% mAP (70.9% v.s. 69.3%). In particular, we achieve 59.4% mAP on COCO object detection, which is the highest accuracy on COCO object detection among DETR-based methods that use the Swin-L model. We achieve 52.38% on nuScenes `val`, which is +1.7% higher than a very recent state-of-the-art approach of PETRv2.

2. Related work

DETR for object detection. With the pioneering work DETR [5] introducing transformers [63] to 2D object detection, more and more follow-up works [11, 15, 50, 68] have built various advanced extensions based on DETR because it removes the need for many hand-designed components like non-maximum suppression [53] or initial anchor boxes generation [17, 35, 38, 57]. Deformable-DETR [89] introduced the multi-scale deformable self/cross-attention scheme, which attends to only a small set of key sampling points around a reference and achieves better performance than DETR (especially on small objects). DAB-DETR [37] further verified that a different novel query formulation can also improve the performance. The follow-up DINO-DETR [24, 82] has established the new SOTA results on object detection tasks and demonstrated the advantages of DETR design by introducing a novel query denoising scheme. Different from these works, we focus on the matching mechanism design of DETR and propose a very simple strategy to improve the training efficacy while still avoiding NMS, which also differentiates our efforts from the very recent DE-DETR [66] that requires NMS.

DETR for other vision tasks. Inspired by the great success of DETR in object detection, many recent efforts have constructed various DETR-based approaches for segmentation tasks [8, 9, 25, 31, 76, 77] that aim for more accurate pixel localization and recognition, 3D object detection tasks [1, 20, 21, 30, 34, 39, 39, 40, 44, 45, 52, 67] based

on point-cloud or multi-view-images that target to identify and localize objects in 3D scenes, pose estimation tasks [2, 26, 28, 48, 59, 60, 83] with the objective of localizing the key points of the presented persons in a given image, object tracking tasks [49, 61, 72, 80, 86] that aim at locating the objects across both spatial and temporal positions within a given video without any prior knowledge about the appearance, and so on. To verify the generalization ability of our approach, we first construct a baseline approach, i.e., Mask-Deformable-DETR, for segmentation tasks, and then combine our hybrid matching scheme with Mask-Deformable-DETR to deliver strong panoptic segmentation results on the COCO benchmark. For 3D object detection, pose estimation, and object tracking, we directly choose the recent approaches including PETRv2 [39, 40], PETR [59], TransTrack [61] as our baseline and verifies that our hybrid matching consistently improves their performance.

Label assignment. We can categorize the existing label assignment approaches, following the previous work [65, 88], into two different paths: (i) *one-to-many label assignment*, i.e., assigning multiple predictions as positive samples for each ground-truth box [57, 62, 84], and (ii) *one-to-one label assignment*, i.e., assigning only one prediction as a positive sample for each ground-truth box. POTO [65] propose to assign the anchor with either the maximum IoU or closest to the object center as the positive sample, which is modified from the strategies of RetinaNet [35] or FCOS [62]. DETR [5] and its followups [4, 24, 37, 50, 68, 89] apply the Hungarian matching to compute one-to-one positive assignments based on the global minimum matching cost values between all predictions and the ground-truth boxes. Different from the most related work POTO [65] that only uses one-to-many assignment, based on ATSS [84], to help the classification branch, our approach chooses Hungarian matching to perform both one-to-one matching and one-to-many matching following DETR and generalizes to various DETR-based approaches across vision tasks.

Relationship to DN-DETR and DINO-DETR: Our approach is also related to recent approaches that introduce noisy augmentations of ground-truth objects as auxiliary queries,
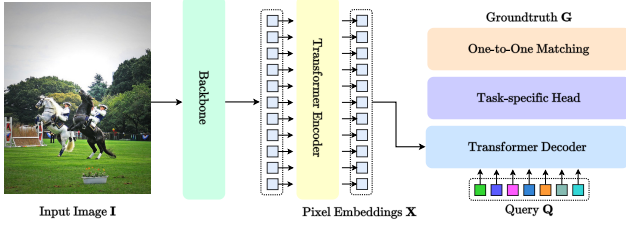
Figure 2. **Illustrating the pipeline of DETR.**

i.e., DN-DETR [24] and DINO-DETR [82]. Similar to our approach, they also introduce additional auxiliary queries. However, the aims of these approaches and ours are different: while DN-DETR and DINO-DETR mainly aim to address the instability issue of Hungarian assignment, we mainly address the insufficient training problem of positive samples in one-to-one matching.

The different aims may have led to their different designs: DN-DETR and DINO-DETR involve a noising scheme and take manual assignment between the noisy queries and ground-truth objects, while our approach is much simpler which uses an end-to-end assignment strategy to match the auxiliary query set the and ground-truth set using the Hungarian method (each ground truth is repeated multiple times). The end-to-end manner also allows our approach to be more general than the methods based on denoising query: while DN-DETR/DINO-DETR needs to tune or redesign its noising scheme and query forms, our approach can be easily extended various DETR variants for different vision problems with almost no additional tuning.

3. Our Approach

3.1. Preliminary

General DETR pipeline. Given an input image $\mathbf{I}$, DETR first applies the backbone and the transformer encoder to extract a sequence of enhanced pixel embeddings $\mathbf{X} = \{\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_N\}$. Second, DETR sends the enhanced pixel embeddings and a default group of object query embeddings $\mathbf{Q} = \{\mathbf{q}_0, \mathbf{q}_1, \cdots, \mathbf{q}_n\}$ into the transformer decoder. Third, DETR applies the task-specific prediction heads on the updated object query embeddings after each transformer decoder layer to generate a set of predictions $\mathbf{P} = \{\mathbf{p}_0, \mathbf{p}_1, \cdots, \mathbf{p}_n\}$ independently. Last, DETR performs one-to-one bipartite matching between the predictions and the ground-truth bounding boxes and labels $\mathbf{G} = \{\mathbf{g}_0, \mathbf{g}_1, \cdots, \mathbf{g}_m\}$. Specifically, DETR associates each ground truth with the prediction that has the minimal matching cost and apply the corresponding supervision accordingly. Figure 2 also illustrates the overall pipeline of the DETR approach. The follow-up works have refor-

mulated the object query to various variants for different visual recognition tasks such as mask query [9, 31], pose query [59], track query [49, 61], bins query [32], and so on. We use "query" in the following discussions for simplicity and consistency.

General Deformable-DETR pipeline. The Deformable-DETR improves the pipeline of DETR from the following three main aspects: (i) replace the original multi-head self-attention or cross-attention with a multi-scale deformable self-attention and multi-scale deformable cross-attention scheme; (ii) replace the original independent layer-wise prediction scheme with iterative refinement prediction scheme; (iii) replace the original image content irrelevantly query with a dynamic query generated by the output from the transformer encoder. Besides, Deformable-DETR also performs one-to-one bipartite matching following the DETR. Readers could refer to [89] for more details.

3.2. Hybrid Matching

The key idea of our hybrid matching approach is to combine the advantages of one-to-one matching scheme with those of the one-to-many matching scheme, where the one-to-one matching is necessary for removing NMS and the one-to-many matching enriches the number of queries that are matched with ground truth for higher training efficacy. We first illustrate detailed implementations of the hybrid branch scheme, and then briefly introduce the implementations of another two simple variants, including the hybrid epoch scheme and hybrid layer scheme. We summarize the pipelines of these hybrid matching schemes in Figure 3.

3.2.1 Hybrid Branch Scheme

We maintain two groups of queries $\mathbf{Q}=\{\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_n\}$ and $\widehat{\mathbf{Q}}=\{\widehat{\mathbf{q}}_1, \widehat{\mathbf{q}}_2, \cdots, \widehat{\mathbf{q}}_T\}$, where we apply one-to-one matching or one-to-many matching on the predictions based on $\mathbf{Q}$ or $\widehat{\mathbf{Q}}$ respectively.

One-to-one matching branch. We process the first group of queries $\mathbf{Q}$ with L transformer decoder layers and perform predictions on the output of each decoder layer respectively. Then, we perform the bipartite matching between the {predictions, ground-truth} pair over each layer, e.g., estimating $\mathcal{L}_{\mathrm{match}}(\mathbf{P}^l, \mathbf{G})$, and compute the loss as follows:

$$\mathcal{L}_{\mathrm{one2one}} = \sum_{l=1}^{L} \mathcal{L}_{\mathrm{Hungarian}}(\mathbf{P}^l, \mathbf{G}), \quad (1)$$

where $\mathbf{P}^l$ represents the predictions outputted by the l-th transformer decoder layer. We choose $\mathcal{L}_{\mathrm{match}}(\cdot)$ and $\mathcal{L}_{\mathrm{Hungarian}}(\cdot)$ following DETR [5] and Deformable-DETR [89], which consist of a classification loss, a $\mathcal{L}_1$ regression loss, and a GIoU loss.
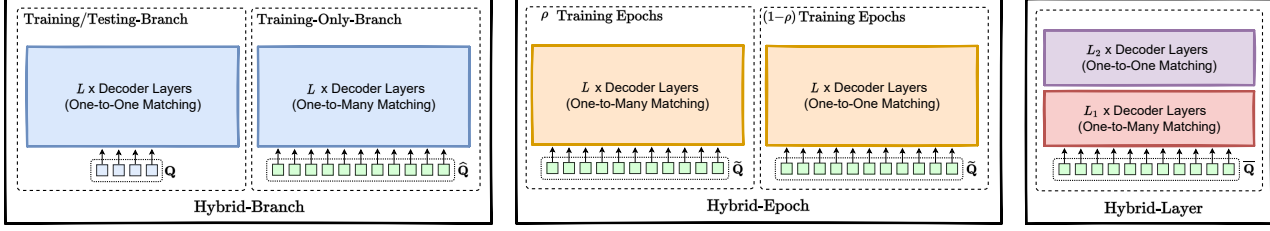
Figure 3. **Illustrating the pipeline of our Hybrid Matching scheme.** We use the colored regions of the same color to mark their parameters are shared. We use ρ to represent the percentage of training epochs. We have $L=L_1 + L_2$ in hybrid layer scheme.

One-to-many matching branch. Then, we process the second group of queries $\widehat{\mathbf{Q}}$ with the same L transformer decoder layers and get L groups of predictions. In order to perform one-to-many matching, we simply repeat the ground truth for K times and get an augmented target $\widehat{\mathbf{G}}=\{\mathbf{G}^1, \mathbf{G}^2, \cdots, \mathbf{G}^K\}$, where $\mathbf{G}^1=\mathbf{G}^2=\cdots=\mathbf{G}^K=\mathbf{G}$. We also perform the bipartite matching between the {predictions, augmented ground truth} pair over each layer, e.g., estimating $\mathcal{L}_{\mathrm{match}}(\widehat{\mathbf{P}}^l, \widehat{\mathbf{G}})$, and compute the corresponding loss as follows:

$$\mathcal{L}_{\mathrm{one2many}} = \sum_{l=1}^{L} \mathcal{L}_{\mathrm{Hungarian}}(\widehat{\mathbf{P}}^l, \widehat{\mathbf{G}}), \qquad (2)$$

where $\widehat{\mathbf{P}}^l$ represents the predictions output by the l-th transformer decoder layer.

In summary, we use the combination of the above two losses, i.e., $\lambda\mathcal{L}_{\mathrm{one2many}}+\mathcal{L}_{\mathrm{one2one}}$, through the whole training process. To accelerate the training speed and process both $\mathbf{Q}$ or $\widehat{\mathbf{Q}}$ in parallel, we further apply a masked multi-head self-attention to avoid their interactions. Therefore, we do not observe significant extra training costs in experiments. We provide detailed comparison results in the following Table 6 within the experiment section. Last, we only keep the one-to-one matching branch, i.e., $\mathbf{Q}$, during evaluation. We present the overall pipeline on the left of Figure 3.

3.2.2 More Variants of Hybrid Matching

Hybrid epoch scheme. Different from the hybrid branch scheme, we only maintain a single group of queries $\widetilde{\mathbf{Q}}=\{\widetilde{\mathbf{q}}_1, \widetilde{\mathbf{q}}_2, \cdots, \widetilde{\mathbf{q}}_M\}$, where we apply both one-to-one matching and one-to-many matching on the predictions based on $\widetilde{\mathbf{Q}}$ during different training epochs. We illustrate more details as follows.

- One-to-many matching training epochs: During the first ρ training epochs, we perform one-to-many matching to process $\widetilde{\mathbf{Q}}$ with L transformer decoder layers and get L groups of predictions. We also get the augmented ground truth $\widetilde{\mathbf{G}}=\{\mathbf{G}^1, \mathbf{G}^2, \cdots, \mathbf{G}^{\widetilde{K}}\}$ following the similar manner adopted by the one-to-many matching branch. Then we perform the bipartite matching between $\mathcal{L}_{\mathrm{match}}(\widetilde{\mathbf{P}}^l, \widetilde{\mathbf{G}})$ and

compute the loss as follows:

$$\mathcal{L}_{\mathrm{one2many}} = \sum_{l=1}^{L} \mathcal{L}_{\mathrm{Hungarian}}(\widetilde{\mathbf{P}}^l, \widetilde{\mathbf{G}}). \qquad (3)$$

- One-to-one matching training epochs: We change one-to-many matching to one-to-one matching for the remaining $(1-\rho)$ training epochs. The only difference is that we match the predictions with the original ground truth and illustrate the formulation as follows:

$$\mathcal{L}_{\mathrm{one2one}} = \sum_{l=1}^{L} \mathcal{L}_{\mathrm{Hungarian}}(\widetilde{\mathbf{P}}^l, \mathbf{G}). \qquad (4)$$

Last, we directly apply $\widehat{\mathbf{Q}}$ during evaluations without using NMS. In summary, we apply only $\mathcal{L}_{\mathrm{one2many}}$ or $\mathcal{L}_{\mathrm{one2one}}$ in the first ρ training epochs or last $(1-\rho)$ training epochs respectively. We also illustrate the overall pipeline of the hybrid epoch scheme in the middle of Figure 3.

Hybrid layer scheme. Similar to the hybrid epoch scheme, we also maintain a single group of queries $\overline{\mathbf{Q}}=\{\overline{\mathbf{q}}_1, \overline{\mathbf{q}}_2, \cdots, \overline{\mathbf{q}}_N\}$. Instead of performing different matching strategies across different training epochs, we apply one-to-many matching on the prediction output by the first L_1 transformer decoder layers and one-to-one matching on the prediction output by the remaining L_2 transformer decoder layers.

- One-to-many matching decoder layers: We choose to apply a one-to-many matching scheme for the first L_1 transformer decoder layers, where we supervise the predictions, output by each one of the first L_1 layers, with the augmented ground truth $\overline{\mathbf{G}}=\{\mathbf{G}^1, \mathbf{G}^2, \cdots, \mathbf{G}^{\overline{K}}\}$ following:

$$\mathcal{L}_{\mathrm{one2many}} = \sum_{l=1}^{L} \mathcal{L}_{\mathrm{Hungarian}}(\overline{\mathbf{P}}^l, \overline{\mathbf{G}}), \qquad (5)$$

where we also need to perform the bipartite matching between $\mathcal{L}_{\mathrm{match}}(\overline{\mathbf{P}}^l, \overline{\mathbf{G}})$ before computing the above loss.

- One-to-one matching decoder layers: For the following L_2 transformer decoder layers, we perform a one-to-one

matching scheme on their predictions as follows:

$$\mathcal{L}_{\text{one2one}} = \sum_{l=L_1}^{L_1+L_2} \mathcal{L}_{\text{Hungarian}}(\overline{\mathbf{P}}^l, \mathbf{G}). \qquad (6)$$

In summary, we apply the combination of both $\mathcal{L}_{\text{one2many}}$ and $\mathcal{L}_{\text{one2one}}$ through the whole training procedure. The right of Figure 3 presents the overall pipeline.

4. Experiment

4.1. Improving DETR-based Approaches

2D object detection results. Table 1 reports the comparison results on the COCO object detection `val` set. Our $\mathcal{H}$-Deformable-DETR achieves consistent gains over the baseline with backbones of different scales (including ResNet-50, Swin-T and Swin-L) trained under 12 epochs or 36 epochs. For example, when choosing Swin-T under 12 and 36 training epochs, our $\mathcal{H}$-Deformable-DETR improves Deformable-DETR from 51.8% to 53.2% and 49.3% to 50.6%, respectively.

Besides, we report the comparison results on LVIS object detection in Table 2. Our approach also achieves consistent gains over the baseline Deformable-DETR across various backbones, e.g., with Swin-L as the backbone, $\mathcal{H}$-Deformable-DETR improves the AP score by $+0.9\%$.

3D object detection results. We choose the very recent representative DETR-based approach, i.e., PETRv2 [40], to verify the generalization ability of our approach for 3D detection based on multi-view images. Table 3 summarizes the detailed comparison results. We can see that our $\mathcal{H}$-PETRv2 significantly improves the NDS scores of baseline PETRv2 from 50.68% to 52.38% on nuScenes `val`, thus showing that our hybrid matching improves the localization accuracy of 3D object detection predictions. We observe the GPU memory consumption increases from 7235M to 11175M, where 78% of the increased GPU memory locates at the cross-attention. A preliminary optimization by sequential self/cross attention [41] has reduced the memory consumption from 11175M to 8733M while maintaining the performance.

Multi-person pose estimation results. We extend our hybrid matching strategy to the very recent PETR (Pose Estimation with TRansformers) [59] and summarize the detailed results in Table 4. We can see that our approach achieves consistent gains over the baselines. For example, with Swin-L as the backbone, our $\mathcal{H}$-PETR improves the AP score of PETR from 73.3% to 74.9% on the COCO `val` under even 100 training epochs.

Multi-object tracking results. We apply our hybrid matching scheme to a powerful multi-object tracking approach,

Table 1. Object detection results on COCO.

method	backbone	#epochs	AP	AP$_S$	AP$_M$	AP$_L$
Results under $1\times$ training schedule						
Deformable-DETR	R50	12	47.0	29.1	50.0	61.6
$\mathcal{H}$-Deformable-DETR	R50	12	**48.7**$^{+1.7}$	31.2	51.5	63.5
Deformable-DETR	Swin-T	12	49.3	31.6	52.4	64.6
$\mathcal{H}$-Deformable-DETR	Swin-T	12	**50.6**$^{+1.3}$	33.4	53.7	65.9
Deformable-DETR	Swin-L	12	54.5	37.0	58.6	71.0
$\mathcal{H}$-Deformable-DETR	Swin-L	12	**55.9**$^{+1.4}$	39.1	59.9	72.2
Results under $3\times$ training schedule						
Deformable-DETR	R50	36	49.0	32.6	52.3	63.3
$\mathcal{H}$-Deformable-DETR	R50	36	**50.0**$^{+1.0}$	32.9	52.7	65.3
Deformable-DETR	Swin-T	36	51.8	34.8	55.1	67.8
$\mathcal{H}$-Deformable-DETR	Swin-T	36	**53.2**$^{+1.4}$	35.9	56.4	68.2
Deformable-DETR	Swin-L	36	56.3	39.2	60.4	71.8
$\mathcal{H}$-Deformable-DETR	Swin-L	36	**57.1**$^{+0.8}$	39.7	61.4	73.4

Table 2. Object detection results on LVIS v1.0.

method	backbone	#epochs	AP	AP$_S$	AP$_M$	AP$_L$
Deformable-DETR	R50	24	32.2	23.2	41.6	49.3
$\mathcal{H}$-Deformable-DETR	R50	24	**33.5**$^{+1.3}$	24.1	42.4	50.2
Deformable-DETR	Swin-L	48	47.0	35.9	57.8	66.9
$\mathcal{H}$-Deformable-DETR	Swin-L	48	**47.9**$^{+0.9}$	36.3	58.6	67.9

e.g., TransTrack [61]. Table 5 summarizes the detailed comparison results. We find the results on MOT17 suffer from relatively large variance, thus we report the mean performance with ~ 3 runs. Accordingly, our approach also shows strong potential on the multi-object tracking tasks, and our $\mathcal{H}$-TransTrack improves the MOTA score of TransTrack from 67.1% to 68.7% on MOT17 `val`, where we also observe that the gains are related to much lower false negative rates (FN).

Panoptic segmentation results. We also report the detailed COCO panoptic segmentation results in the supplementary to verify the generalization ability of our approach.

4.2. Ablation Study

Comparing different hybrid matching schemes. We first choose a two-stage Deformable-DETR (with increased FFN dimension) as the baseline and compare the results based on our three different hybrid matching approaches. To ensure fairness, we choose their settings as follows:

- *Baseline settings*: We use 300 or 1800 queries and apply the conventional one-to-one matching following the original Deformable-DETR [89]. To ensure more fair comparisons with our hybrid branch scheme, we further report the baseline results under $1.3\times$ and $1.15\times$ training epochs to

Table 3. Multi-view 3D detection results on nuScenes.

method	backbone	#epochs	mAP	NDS
PETRv2 [40]	VoVNet-99	24	41.04	50.25
PETRv2 (Our repro.)	VoVNet-99	24	40.41	49.69
$\mathcal{H}$-PETRv2	VoVNet-99	24	$\textbf{41.93}^{+1.52}$	51.23
PETRv2 (Our repro.)	VoVNet-99	36	41.07	50.68
$\mathcal{H}$-PETRv2	VoVNet-99	36	$\textbf{42.59}^{+1.52}$	52.38

Table 4. Multi-person pose estimation results on COCO.

method	backbone	#epochs	AP	AP_M	AP_L
PETR [59]	R50	100	68.8	62.7	77.7
PETR (Our repro.)	R50	100	69.3	63.3	78.4
$\mathcal{H}$-PETR	R50	100	$\textbf{70.9}^{+1.6}$	64.4	80.3
PETR [59]	R101	100	70.0	63.6	79.4
PETR (Our repro.)	R101	100	69.9	63.4	79.4
$\mathcal{H}$-PETR	R101	100	$\textbf{71.0}^{+1.1}$	64.7	80.2
PETR [59]	Swin-L	100	73.1	67.2	81.7
PETR (Our repro.)	Swin-L	100	73.3	67.7	81.6
$\mathcal{H}$-PETR	Swin-L	100	$\textbf{74.9}^{+1.6}$	69.3	83.3

Table 5. Multi-object tracking results on MOT.

method	#epochs	MOTA ($\uparrow$)	IDF1 ($\uparrow$)	FN ($\downarrow$)
MOT17 `val`				
TransTrack [61]	20	67.1	70.3	15820
TransTrack (Our repro.)	20	67.1	68.1	15680
$\mathcal{H}$-TransTrack	20	$\textbf{68.7}^{+1.6}$	68.3	13657
MOT17 `test`				
TransTrack [61]	20	74.5	63.9	112137
TransTrack (Our repro.)	20	74.6	63.2	111105
$\mathcal{H}$-TransTrack	20	$\textbf{75.7}^{+1.1}$	64.4	91155

ensure even more total training time than our hybrid branch scheme. ==We empirically find that the increased time mainly comes from two aspects including more queries (1800 vs. 300) and the extra Hungarian matching operation and loss computation of one-to-many matching branch.== By simply merging the matching cost computation and loss computation of one-to-one matching branch and one-to-many matching branch, our hybrid-branch scheme only bring around +7% extra training time compared to the baseline with 1800 queries. The extra time could be further decreased by implementing the Hungarian matching on GPU instead of CPU, which is not the focus of this work.

- *Hybrid branch settings*: We use 300 queries (n=300) and 1500 queries (T=1500) for one-to-one matching and one-to-many matching branch, respectively. We set K as 6 for the augmented ground truth $\widehat{G}$. Thus we will generate $6\times$ additional positive queries with the one-to-many matching branch, e.g., we have $(1+6)\times12\times6$ under 12 training epochs if the benchmark is the number of positive queries within one training epoch for each decoder layer.

Table 6. Comparisons of different hybrid matching approach. We ensure that three different hybrid approaches all (i) introduce $6\times$ more positive samples than the baseline, and (ii) use 1800 query in total. The upper script † marks the methods using 1800 query during both training and evaluation. The time is averaged over all training epochs as a hybrid epoch scheme consists of different training stages. The FPS is tested on the same V100 GPU. The upper script ♮ marks the time measured with the optimized implementation and we provide more details in the supplementary.

| method	inference GFLOPs	train time (average)	inference FPS	#epochs		
				12	24	36
Deformable-DETR	**268G**	**65**min	**6.7**	43.7	46.4	46.8
Deformable-DETR$^{1.3\times}$	**268G**	**65**min	**6.7**	44.6	46.3	46.7
Deformable-DETR†	282G	75min	6.3	44.1	46.6	47.1
Deformable-DETR$^{\dagger 1.15\times}$	282G	75min	6.3	44.5	46.7	46.9
Deformable-DETR + Hybrid-Branch	**268G**	80min$^{\natural}$	**6.7**	45.9	47.6	**48.0**
Deformable-DETR + Hybrid-Epoch†	282G	95min	6.3	45.5	47.0	47.9
Deformable-DETR + Hybrid-Layer†	282G	100min	6.3	45.6	47.9	**48.0**
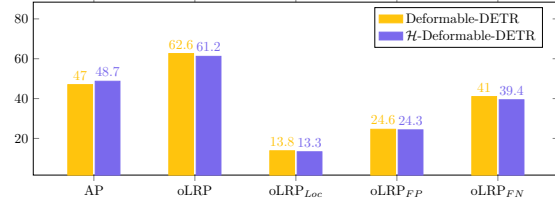
Figure 4. Illustrating the improvements of hybrid matching over Deformable-DETR. We report both average precision (AP) and optimal localization recall precision (oLRP [54]) scores, where the higher the better for AP and the lower the better for oLRP. Our approach significantly decreases $oLRP_{Loc}$ and $oLRP_{FN}$, which represent the localization error of matched detections (true positives) and the number of unmatched ground truth (false negatives).

- *Hybrid epoch settings*: We use 1800 queries through the whole training process, i.e., M=1800. To generate $6\times$ additional positive queries than the baseline, we set $\widetilde{K}$=10 and $\rho=\frac{2}{3}$. Therefore, we have $10\times\frac{2}{3}\times12\times6+\frac{1}{3}\times12\times6=(1+6)\times12\times6$ holds under 12 training epochs.

- *Hybrid layer settings*: We also use 1800 queries through the whole training process, i.e., N=1800. To generate $6\times$ additional positive queries than the baseline, we set $\overline{K}$=10, L_1=4, and L_2=2 considering that $10\times12\times4+12\times2=(1+6)\times12\times6$ holds.

Table 6 summarizes the detailed comparison results. Accordingly, we find that hybrid matching approaches consistently outperform the baseline under different training epochs (including $1.3\times$ and $1.15\times$ training epochs). We also report the GPU memory of hybrid-branch/hybrid-layer/hybrid epoch (for ResNet50): 7728M/7722M/7711M. We can see that the hybrid branch scheme is the best choice if we consider the trade-off of training time, inference

Table 7. GFLOPs/training time/GPU memory vs. # query based on our hybrid branch scheme. The numbers within () are based on Swin-L.

backbone	method	# query [hyper-parameter]	GFLOPs	training time (min)	GPU memory (M)
ResNet50 (Swin-L)	Baseline	300 [n=300,T=0,K=0]	268.19 (912.29)	65 (202)	5480 (8955)
	Ours	600 [n=300,T=300,K=6]	271.24 (915.34)	71 (205)	5719 (9190)
		900 [n=300,T=600,K=6]	274.03 (918.12)	72 (208)	6045 (9530)
		1200 [n=300,T=900,K=6]	276.82 (920.91)	75 (210)	6528 (10006)
		1500 [n=300,T=1200,K=6]	279.60 (923.69)	78 (213)	7071 (10558)
		1800 [n=300,T=1500,K=6]	282.39 (926.48)	80 (215)	7728 (11203)

GFLOPs/FPS, and accuracy. Therefore, we choose the hybrid branch scheme (for its faster training and inference) by default if not specified in the following ablation experiments.

About computation/training time/GPU memory cost. Although $\mathcal{H}$-Deformable-DETR uses $6\times$ number of queries, it has small training overhead on the original Deformable-DETR framework: $1.6\%/5.4\%$ on GFLOPs and $6.4\%/23.1\%$ on training times for Swin-L/ResNet50, respectively (see Table 7). This is because only a portion of the decoder computation is affected by the number of queries, which takes about $0.3\%/1.1\%$ of the entire network FLOPs for Swin-L/ResNet50. Also note that the more increases in training time is caused by the Hungarian matching (implemented on CPU) and the cost/loss computation steps, which are inessential, and we leave its optimization as our future work.

Most of the GPU memory consumption overhead by our method comes from the naive self/cross-attention modules. The large memory complexity is not intrinsic, and it can be significantly reduced by advanced implementation, e.g., Flash Attention [12] can decrease the memory to linear complexity, and the sequential attention technique in SwinV2 can also greatly reduce the memory consumption for both self/cross attention. Both implementations are equivalent to the original one, thus not compromising accuracy. With these optimized implementation, the memory consumption increases by more queries are mild.

Effect of each component based on Deformable-DETR. We choose two-stage Deformable-DETR as our baseline and report the detailed improvements of each component in Table 8. Notably, we do not observe the obvious benefits of applying drop-out rate as zero, mixed query selection, and looking forward twice[1] in other tasks. Therefore, we only apply this combination of tricks on 2D object detection tasks by default if not specified. We further analyze the detailed improvements of our approach based on the metrics of optimal localization recall precision in Figure 4, where the lower the better. In summary, we can see that our approach mainly improves the performance from two aspects, including more accurate localization and fewer false negatives.

---

[1]We implement both mixed query selection and looking forward twice following DINO [82].

Table 8. Comparison results based on two-stage Deformable-DETR on COCO 2017 `val` under 12 training epochs. $2\times$FFN: increase FFN dimension from $1,024$ to $2,048$. DP0: setting the drop out rate within transformer as 0. MQS: mixed query selection. LFT: look forward twice. HM: our hybrid matching.

$2\times$FFN	DP0	MQS	LFT	HM	AP
✗	✗	✗	✗	✗	43.3
✓	✗	✗	✗	✗	43.7
✓	✓	✗	✗	✗	44.3
✓	✓	✓	✗	✗	46.3
✓	✓	✓	✓	✗	47.0
✓	✓	✓	✓	✓	**48.7**

Table 9. Influence of K of our approach on COCO 2017 `val` under 12 training epochs. We set $T = 300 \times K$.

K	0	1	2	3	4	5	6	7	8
AP	47.0	46.4	46.7	48.1	48.4	48.3	**48.6**	48.5	**48.6**

Table 10. Influence of T of our approach on COCO 2017 `val` under 12 training epochs. We set $K = 6$.

T	300	600	900	1200	1500	1800
AP	47.8	48.3	48.4	48.4	**48.7**	48.6

Choice of K within the one-to-many matching branch. We study the influence of the choice of K in Table 9. Accordingly, we find our approach achieves consistent gains only when choosing K larger than 3. We choose K as 6 on COCO by default. We guess the reasons for the performance drops with smaller K and increases with larger K are: low-quality queries harm accuracy while larger K benefit auxiliary loss which helps accuracy. We verify the quality of different groups of queries between 300-1800 by training a new detector that replaces the original proposals of query 0-300. The APs of queries 300-600,600-900,900-1200,1200-1500,1500-1800 are $42.7/42.8/43.0/43.1/42.5$, suggesting that different groups have similar qualities, although lower than the default of 0-300 (AP=47.0). This indicates the low-quality issue will not get more serious when increasing K, while the auxiliary loss can benefit more from larger K. We also show that designing a careful selection mechanism for the one-to-many matching branch can achieve consistent gains even when using K=1 but do not observe any further benefits when using large K values. More details are summarized in the supplementary.

Choice of T within the one-to-many matching branch. We study the influence of the total number of queries T within the one-to-many matching branch by fixing K as 6 in Table 10. We choose T=1500 on the COCO object detection task as it achieves the best results.

Effect of sharing parameters. We study the influence of the sharing parameters across the one-to-one matching branch and one-to-many matching branch in Table 12. We can observe that (i) using independent classification heads

Table 11. System-level comparisons with the leading single-scale evaluation results on COCO `val`.

method	framework	backbone	input size	#epochs	AP	AP$_{50}$	AP$_{75}$	AP$_S$	AP$_M$	AP$_L$
Swin [42]	HTC	Swin-L	1600×1200	36	57.1	75.6	62.5	42.4	60.7	71.1
CBNetV2 [33]	HTC	$2\times$ Swin-L	1600×1400	12	59.1	-	-	-	-	-
ConvNeXt [43]	Cascade Mask R-CNN	ConvNeXt-XL	1333×800	36	55.2	74.2	59.9	-	-	-
MViTv2 [27]	Cascade Mask R-CNN	MViTv2-L	1333×800	50	55.8	74.3	64.3	-	-	-
MOAT [73]	Cascade Mask R-CNN	MOAT-3	1344×1344	36	59.2	**77.8**	60.9	-	-	-
Group-DETR [6]	DETR	Swin-L	1333×800	36	58.4	-	-	41.0	62.5	73.9
DINO-DETR [82]	DETR	Swin-L	1333×800	36	58.5	77.0	64.1	41.5	62.3	74.0
$\mathcal{H}$-Deformable-DETR	DETR	Swin-L	1333×800	36	**59.4**	**77.8**	**65.4**	**43.1**	**63.1**	**74.2**

Table 12. Influence of sharing parameters on COCO 2017 `val` under 12 training epochs.

trans. encoder	trans. decoder	box head	cls head	AP
✓	✓	✓	✓	**48.7**
✓	✓	✓	✗	48.6
✓	✓	✗	✗	48.5
✓	✗	✗	✗	48.3
✗	✗	✗	✗	47.3

Table 13. Comparison to only using one-to-many matching.

method	NMS	train time (average)	inference FPS	#epochs		
				12	24	36
Only one-to-many matching	✓	95min	5.3	**49.4**	**50.2**	48.8
Ours (one-to-one branch)	✗		6.7	48.7	49.9	**50.0**
Ours (one-to-one branch)	✓	80min	5.6	48.7	50.0	**50.0**
Ours (one-to-many branch)	✗		6.5	13.5	13.1	12.9
Ours (one-to-many branch)	✓		5.4	48.6	49.8	49.9

or bounding box heads does not hurt the performance, (ii) further using independent transformer decoder layers slightly drops, and (iii) further using independent transformer encoder layers results in significant performance drops considering the baseline performance is 47.0%. Accordingly, we can see that the better optimization of transformer encoder instead of decoder is the key to the performance improvements.

To verify whether this observation still holds on DINO-DETR, we also report the detailed ablation results in the supplementary to verify that most of the gains of (contrast) query denoising essentially come from the better optimization of transformer encoder. For example, using independent transformer decoders and independent box & classification heads only suffers from 0.1% ↓ drop while further using an independent transformer encoder contributes to another 0.5% ↓ drop.

Comparison to only using one-to-many matching. Table 13 compares our approach to a strong variant that applies only one-to-many matching through the whole training process and NMS during evaluation. We also ablate the predictions based only on the one-to-one matching branch

or one-to-many matching branch in Table 13. Accordingly, we see that (i) the one-to-many branch evaluated with NMS achieves comparable performance with the one-to-one matching branch; (ii) only using one-to-many matching evaluated with NMS achieves slightly better performance than ours; (ii) the one-to-one matching branch within our hybrid matching scheme is the best choice when considering the training time and inference speed (FPS).

We further conduct more ablation experiments including: analyzing the training/validation loss curves of the one-to-one matching branch, the influence of loss-weight on one-to-many matching branch, effect of a careful query selection scheme, precision-recall curves, and so on.

4.3. Comparison with State-of-the-art

Table 11 reports the system-level comparisons to some representative state-of-the-art methods that use single-scale evaluation on COCO `val` and choose backbones of similar capacity with Swin-L. By introducing additional enhancement techniques in [82], our $\mathcal{H}$-Deformable-DETR achieves 59.4% on COCO `val` set, which surpasses the very recent DINO-DETR method, as well as other top-performing methods.

5. Conclusion

This paper presents a very simple yet surprisingly effective hybrid matching scheme to address the low training efficacy of DETR-based approaches on multiple vision tasks. Our approach explicitly combines the advantages of a one-to-one matching scheme, i.e., avoiding NMS, and those of a one-to-many matching scheme, i.e., increasing the number of positive queries and training efficacy. We hope our initial efforts can accelerate the advancement of DETR approaches on various vision tasks.

References

[1] Xuyang Bai, Zeyu Hu, Xinge Zhu, Qingqiu Huang, Yilun Chen, Hongbo Fu, and Chiew-Lan Tai. Transfusion: Robust lidar-camera fusion for 3d object detection with transformers. In *CVPR*, 2022. 1, 2

[2] Guillem Brasó, Nikita Kister, and Laura Leal-Taixé. The center of attention: Center-keypoint grouping via attention for multi-person pose estimation. In *ICCV*, 2021. 2

[3] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 12

[4] Xipeng Cao, Peng Yuan, Bailan Feng, Kun Niu, and Yao Zhao. Cf-detr: Coarse-to-fine transformers for end-to-end object detection. In *AAAI*, 2022. 1, 2

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

[6] Qiang Chen, Xiaokang Chen, Gang Zeng, and Jingdong Wang. Group detr: Fast training convergence with decoupled one-to-many label assignment. *arXiv:2207.13085*, 2022. 8

[7] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *CVPR*, 2021. 1

[8] Bowen Cheng, Anwesa Choudhuri, Ishan Misra, Alexander Kirillov, Rohit Girdhar, and Alexander G Schwing. Mask2former for video instance segmentation. *arXiv:2112.10764*, 2021. 1, 2

[9] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. *arXiv:2112.01527*, 2021. 1, 2, 3

[10] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 12

[11] Xiyang Dai, Yinpeng Chen, Jianwei Yang, Pengchuan Zhang, Lu Yuan, and Lei Zhang. Dynamic detr: End-to-end object detection with dynamic attention. In *ICCV*, 2021. 2

[12] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *NeurIPS*, 2022. 7

[13] Bin Dong, Fangao Zeng, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. Solq: Segmenting objects by learning queries. *NeurIPS*, 2021. 1

[14] Yuxin Fang, Shusheng Yang, Xinggang Wang, Yu Li, Chen Fang, Ying Shan, Bin Feng, and Wenyu Liu. Instances as queries. In *ICCV*, 2021. 1

[15] Peng Gao, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fast convergence of detr with spatially modulated co-attention. In *ICCV*, 2021. 1, 2

[16] Ziteng Gao, Limin Wang, Bing Han, and Sheng Guo. Adamixer: A fast-converging query-based object detector. In *CVPR*, 2022. 1

[17] Ross Girshick. Fast r-cnn. In *ICCV*, 2015. 2

[18] Brent A Griffin and Jason J Corso. Depth from camera motion and object detection. In *CVPR*, 2021. 1

[19] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019. 12

[20] Junjie Huang and Guan Huang. Bevdet4d: Exploit temporal cues in multi-camera 3d object detection. *arXiv:2203.17054*, 2022. 2

[21] Junjie Huang, Guan Huang, Zheng Zhu, and Dalong Du. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. *arXiv:2112.11790*, 2021. 2

[22] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. Mdetr-modulated detection for end-to-end multi-modal understanding. In *ICCV*, 2021. 1

[23] Youngwan Lee and Jongyoul Park. Centermask: Real-time anchor-free instance segmentation. In *CVPR*, 2020. 2

[24] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M Ni, and Lei Zhang. Dn-detr: Accelerate detr training by introducing query denoising. *arXiv:2203.01305*, 2022. 1, 2, 3

[25] Feng Li, Hao Zhang, Shilong Liu, Lei Zhang, Lionel M Ni, Heung-Yeung Shum, et al. Mask dino: Towards a unified transformer-based framework for object detection and segmentation. *arXiv:2206.02777*, 2022. 1, 2

[26] Ke Li, Shijie Wang, Xiang Zhang, Yifan Xu, Weijian Xu, and Zhuowen Tu. Pose recognition with cascade transformers. In *CVPR*, 2021. 1, 2

[27] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Improved multiscale vision transformers for classification and detection. *arXiv:2112.01526*, 2021. 8

[28] Yanjie Li, Shoukui Zhang, Zhicheng Wang, Sen Yang, Wankou Yang, Shu-Tao Xia, and Erjin Zhou. Tokenpose: Learning keypoint tokens for human pose estimation. In *ICCV*, 2021. 2

[29] Zhenyu Li, Zehui Chen, Xianming Liu, and Junjun Jiang. Depthformer: Exploiting long-range correlation and local information for accurate monocular depth estimation. *arXiv:2203.14211*, 2022. 1

[30] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. *arXiv:2203.17270*, 2022. 1, 2

[31] Zhiqi Li, Wenhai Wang, Enze Xie, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, Ping Luo, and Tong Lu. Panoptic segformer: Delving deeper into panoptic segmentation with transformers. In *CVPR*, 2022. 1, 2, 3

[32] Zhenyu Li, Xuyang Wang, Xianming Liu, and Junjun Jiang. Binsformer: Revisiting adaptive bins for monocular depth estimation. *arXiv:2204.00987*, 2022. 3

[33] Tingting Liang, Xiaojie Chu, Yudong Liu, Yongtao Wang, Zhi Tang, Wei Chu, Jingdong Chen, and Haibin Ling. Cbnet: A composite backbone network architecture for object detection. *TIP*, 2022. 8

[34] Tingting Liang, Hongwei Xie, Kaicheng Yu, Zhongyu Xia, Zhiwei Lin, Yongtao Wang, Tao Tang, Bing Wang, and Zhi

Tang. Bevfusion: A simple and robust lidar-camera fusion framework. *arXiv:2205.13790*, 2022. 2

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

[36] 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 *ECCV*, 2014. 12

[37] 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:2201.12329*, 2022. 1, 2

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

[39] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. *arXiv:2203.05625*, 2022. 2

[40] Yingfei Liu, Junjie Yan, Fan Jia, Shuailin Li, Qi Gao, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petrv2: A unified framework for 3d perception from multi-camera images. *arXiv:2206.01256*, 2022. 1, 2, 5, 6

[41] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *CVPR*, 2022. 5

[42] 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 *ICCV*, 2021. 8

[43] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *CVPR*, 2022. 8

[44] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. *arXiv:2205.13542*, 2022. 2

[45] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. In *ICCV*, 2021. 2

[46] Shangbang Long, Siyang Qin, Dmitry Panteleev, Alessandro Bissacco, Yasuhisa Fujii, and Michalis Raptis. Towards end-to-end unified scene text detection and layout analysis. In *CVPR*, 2022. 1

[47] Qian Lou, Yen-Chang Hsu, Burak Uzkent, Ting Hua, Yilin Shen, and Hongxia Jin. Lite-mdetr: A lightweight multimodal detector. In *CVPR*, 2022. 1

[48] Weian Mao, Yongtao Ge, Chunhua Shen, Zhi Tian, Xinlong Wang, and Zhibin Wang. Tfpose: Direct human pose estimation with transformers. *arXiv:2103.15320*, 2021. 2

[49] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. In *CVPR*, 2022. 1, 2, 3

[50] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *ICCV*, 2021. 1, 2

[51] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *arXiv:1603.00831*, 2016. 12

[52] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. In *ICCV*, 2021. 1, 2

[53] Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *ICPR*, 2006. 2

[54] Kemal Oksuz, Baris Can Cam, Sinan Kalkan, and Emre Akbas. One metric to measure them all: Localisation recall precision (lrp) for evaluating visual detection tasks. *TPAMI*, 2021. 6

[55] Zobeir Raisi, Mohamed A Naiel, Georges Younes, Steven Wardell, and John S Zelek. Transformer-based text detection in the wild. In *CVPR*, 2021. 1

[56] Zobeir Raisi, Georges Younes, and John Zelek. Arbitrary shape text detection using transformers. *arXiv:2202.11221*, 2022. 1

[57] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NeurIPS*, 2015. 2

[58] Byungseok Roh, JaeWoong Shin, Wuhyun Shin, and Saehoon Kim. Sparse detr: Efficient end-to-end object detection with learnable sparsity. *arXiv:2111.14330*, 2021. 1

[59] Dahu Shi, Xing Wei, Liangqi Li, Ye Ren, and Wenming Tan. End-to-end multi-person pose estimation with transformers. In *CVPR*, 2022. 1, 2, 3, 5, 6

[60] Lucas Stoffl, Maxime Vidal, and Alexander Mathis. End-to-end trainable multi-instance pose estimation with transformers. *arXiv:2103.12115*, 2021. 1, 2

[61] Peize Sun, Jinkun Cao, Yi Jiang, Rufeng Zhang, Enze Xie, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple object tracking with transformer. *arXiv:2012.15460*, 2020. 1, 2, 3, 5, 6, 12

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

[63] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2

[64] Huiyu Wang, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Max-deeplab: End-to-end panoptic segmentation with mask transformers. In *CVPR*, 2021. 1

[65] Jianfeng Wang, Lin Song, Zeming Li, Hongbin Sun, Jian Sun, and Nanning Zheng. End-to-end object detection with fully convolutional network. In *CVPR*, 2021. 1, 2

[66] Wen Wang, Jing Zhang, Yang Cao, Yongliang Shen, and Dacheng Tao. Towards data-efficient detection transformers. *arXiv:2203.09507*, 2022. 2

[67] Yue Wang, Vitor Campagnolo Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, and Justin Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In *CoRL*, 2022. 1, 2

[68] Yingming Wang, Xiangyu Zhang, Tong Yang, and Jian Sun. Anchor detr: Query design for transformer-based detector. *arXiv:2109.07107*, 2021. 1, 2

[69] Jiannan Wu, Yi Jiang, Peize Sun, Zehuan Yuan, and Ping Luo. Language as queries for referring video object segmentation. *arXiv:2201.00487*, 2022. 1

[70] Junfeng Wu, Yi Jiang, Wenqing Zhang, Xiang Bai, and Song Bai. Seqformer: a frustratingly simple model for video instance segmentation. *arXiv:2112.08275*, 2021. 1

[71] Yifan Xu, Weijian Xu, David Cheung, and Zhuowen Tu. Line segment detection using transformers without edges. In *CVPR*, 2021. 1

[72] Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. Learning spatio-temporal transformer for visual tracking. In *ICCV*, 2021. 2

[73] Chenglin Yang, Siyuan Qiao, Qihang Yu, Xiaoding Yuan, Yukun Zhu, Alan Yuille, Hartwig Adam, and Liang-Chieh Chen. Moat: Alternating mobile convolution and attention brings strong vision models. *arXiv:2210.01820*, 2022. 8

[74] Zhao Yang, Jiaqi Wang, Yansong Tang, Kai Chen, Hengshuang Zhao, and Philip HS Torr. Lavt: Language-aware vision transformer for referring image segmentation. In *CVPR*, 2022. 1

[75] Qihang Yu, Huiyu Wang, Dahun Kim, Siyuan Qiao, Maxwell Collins, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Cmt-deeplab: Clustering mask transformers for panoptic segmentation. In *CVPR*, 2022. 1

[76] Qihang Yu, Huiyu Wang, Dahun Kim, Siyuan Qiao, Maxwell Collins, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Cmt-deeplab: Clustering mask transformers for panoptic segmentation. In *CVPR*, 2022. 2

[77] Qihang Yu, Huiyu Wang, Siyuan Qiao, Maxwell Collins, Yukun Zhu, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. k-means mask transformer. In *ECCV*, 2022. 2

[78] Xiaodong Yu, Dahu Shi, Xing Wei, Ye Ren, Tingqun Ye, and Wenming Tan. Soit: Segmenting objects with instance-aware transformers. *arXiv:2112.11037*, 2021. 1

[79] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. In *ECCV*, 2020. 1

[80] Fangao Zeng, Bin Dong, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. Motr: End-to-end multiple-object tracking with transformer. *arXiv:2105.03247*, 2021. 2

[81] Gongjie Zhang, Zhipeng Luo, Yingchen Yu, Kaiwen Cui, and Shijian Lu. Accelerating DETR convergence via semantic-aligned matching. In *CVPR*, 2022. 1

[82] 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:2203.03605*, 2022. 1, 2, 3, 7, 8

[83] Jianfeng Zhang, Yujun Cai, Shuicheng Yan, Jiashi Feng, et al. Direct multi-view multi-person 3d pose estimation. *NeurIPS*, 2021. 2

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

[85] Xiang Zhang, Yongwen Su, Subarna Tripathi, and Zhuowen Tu. Text spotting transformers. In *CVPR*, 2022. 1

[86] Moju Zhao, Kei Okada, and Masayuki Inaba. Trtr: Visual tracking with transformer. *arXiv:2105.03817*, 2021. 2

[87] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Phillip Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. *arXiv:2201.02605*, 2022. 12

[88] Benjin Zhu, Jianfeng Wang, Zhengkai Jiang, Fuhang Zong, Songtao Liu, Zeming Li, and Jian Sun. Autoassign: Differentiable label assignment for dense object detection. *arXiv:2007.03496*, 2020. 2

[89] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv:2010.04159*, 2020. 1, 2, 3, 5

Supplementary

A. Datasets

COCO [36]. The COCO object detection dataset consists of 123K images with 896K annotated bounding boxes belonging to 80 thing classes and 53 stuff classes, where the `train` set contains 118K images and the `val` set contains 5K images. We report the 2D object detection performance on the `val` set. The COCO pose estimation benchmark consists of more than 200K images and 250K person instances labeled with 17 keypoints, where `train` set consists of 57K images and 150K person instances, `val` set consists of 5K images, and `test-dev` set consists of 20K images, respectively.

LVIS [19]. This dataset consists of 100K images annotated with both object detection bounding boxes and instance segmentation masks for 1203 classes. We follow the very recent work Detic [87] to only use the bounding box supervision for training.

nuScenes [3]. This is a large-scale autonomous driving dataset consisting of 1000 driving sequences with each one about 20s long, where the multimodel dataset is collected from 6 cameras, 1 lidar, and 5 radars. We partition the 1000 driving sequences to 700, 150, and 150 sequences for `train`, `val`, and `test` respectively. We report both the numbers of nuScenes Detection Score (NDS) and mean Average Precision (mAP) to measure the 3D object detection performance.

ScanNetV2 [10]. This dataset consists of 1513 indoor scenes labeled with per-point instance ids, semantic categories and 3D bounding boxes for around 18 categories. We use 1201 and 312 scenes for `train` and `val`, respectively. We mainly report the mAP scores under two different IoU thresholds, i.e., 0.25 and 0.5.

MOT17 [51]. This dataset consists of 14 pedestrian tracking videos annotated with rich bounding boxes and their corresponding track ids. We use 7 videos for `train` and the other 7 videos for `test`. Following TransTrack [61], we split the second half of each `train` video to form a `val` set. We report the multi-object tracking performance on both `val` and `test`.

B. More Hyper-parameter Details

We illustrate the detailed hyper-parameter settings when applying our hybrid branch approach to different DETR-based approaches and different benchmarks in Table 14.

C. Panoptic Segmentation Results

To verify the effectiveness of our approach to the panoptic segmentation task, we first construct a simple baseline, i.e., Mask-Deformable-DETR, by adding a mask prediction

Table 14. Illustrating the hyper-parameter settings. $2\times$FFN: increase FFN dimension from $1,024$ to $2,048$. DP0: setting the drop out rate within transformer as 0. MQS: mixed query selection. LFT: look forward twice.

Method	$2\times$FFN	DP0	MQS	LFT	Dataset	n	T	K	λ
$\mathcal{H}$-Deformable-DETR	✓	✓	✓	✓	COCO	300	1500	6	1.0
					LVIS	300	900	5	1.0
$\mathcal{H}$-PETR	✓	✓	✗	✗	COCO	300	900	5	1.0
$\mathcal{H}$-PETRv2	✗	✗	✗	✗	nuScenes	900	1800	4	1.0
$\mathcal{H}$-TransTrack	✗	✗	✗	✗	MOT17	500	1000	5	0.5

Table 15. Panoptic segmentation results on COCO `val`.

Model	Backbone	#epochs	PQ
Mask-Deformable-DETR	R50	12	47.0
$\mathcal{H}$-Mask-Deformable-DETR	R50	12	$\mathbf{48.5}^{+1.5}$
Mask-Deformable-DETR	R50	50	51.5
$\mathcal{H}$-Mask-Deformable-DETR	R50	50	$\mathbf{52.1}^{+0.6}$
Mask-Deformable-DETR	R50	100	52.2
$\mathcal{H}$-Mask-Deformable-DETR	R50	100	$\mathbf{52.6}^{+0.4}$
Mask-Deformable-DETR	Swin-T	50	52.9
$\mathcal{H}$-Mask-Deformable-DETR	Swin-T	50	$\mathbf{53.5}^{+0.6}$
Mask-Deformable-DETR	Swin-T	100	53.8
$\mathcal{H}$-Mask-Deformable-DETR	Swin-T	100	$\mathbf{54.2}^{+0.4}$
Mask-Deformable-DETR	Swin-L	50	56.7
$\mathcal{H}$-Mask-Deformable-DETR	Swin-L	50	$\mathbf{57.0}^{+0.3}$
Mask-Deformable-DETR	Swin-L	100	56.9
$\mathcal{H}$-Mask-Deformable-DETR	Swin-L	100	$\mathbf{57.2}^{+0.3}$

head to the Deformable-DETR. Then, we apply our hybrid branch scheme to the Mask-Deformable-DETR following the $\mathcal{H}$-Deformable-DETR, thus resulting in $\mathcal{H}$-Mask-Deformable-DETR. We summarize the detailed comparison results in Table 15. Accordingly, we can see that our hybrid matching scheme consistently improves ResNet50, Swin-T, and Swin-L under various training epochs.

For example, $\mathcal{H}$-Mask-Deformable-DETR gains $+0.3\%$ over Mask-Deformable-DETR equipped with Swin-L when trained for 50 epochs. We guess the relatively limited gains when compared to object detection tasks, are due to that *the transformer encoder receives much more dense and informative training signals from the extra introduced mask branch*, which consists of the dense interactions between the query embeddings and the high-resolution feature maps output by the transformer encoder. We also have verified that better training of the transformer encoder is the key factor to the performance improvements.

D. More Ablation Results

- First, we illustrate the curves of training losses and validation losses based on Deformable-DETR in Figure 5. We
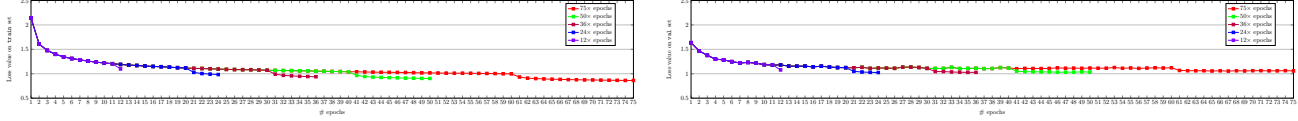
Figure 5. Illustrating the loss curves of Deformable-DETR on `train` and `val`. We can see that, with longer training epochs, e.g., from 50 epochs to 75 epochs, the training loss consistently decreases while the validation loss saturates on COCO object detection benchmark.
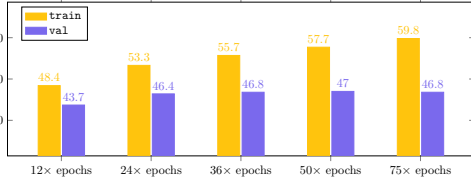

Figure 6. Illustrating the AP scores of Deformable-DETR on `train` and `val` under longer training epochs. We can see that, with longer training epochs, e.g., from 50 epochs to 75 epochs, the AP scores on `train` set consistently improves while the AP scores on `val` set saturates on COCO object detection benchmark.

Table 16. Influence of λ of our approach on COCO 2017 `val` under $12\times$ epochs training schedule. We set $K = 6$ and $T = 1500$.

λ	0.1	0.2	0.5	1	2	5
AP	47.8	48.0	48.4	**48.7**	48.5	48.3

Table 17. Influence of sharing parameters for DINO framework.

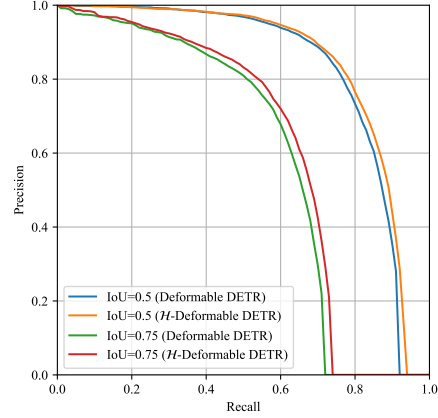| trans. encoder	trans. decoder	box head	cls head	AP
✓	✓	✓	✓	49.1
✓	✓	✓	✗	48.9$^{-0.2}$
✓	✓	✗	✗	49.2$^{+0.1}$
✓	✗	✗	✗	49.0$^{-0.1}$
✗	✗	✗	✗	48.5$^{-0.6}$

Figure 7. Illustrating the Precision-Recall curves. We can see that our approach consistently improves the recall of the baseline under two different IoU thresholds

also report the detection performance of Deformable-DETR under various training schedules from $12\times$ epochs to $75\times$ epochs in Figure 6. Accordingly, we observe that (i) simply increasing the number of training epochs fails to improve the performance, and (ii) Deformable-DETR can not benefit from exploring more positive queries that are matched with ground truth during the additional training epochs, which further verifies the advantage of our hybrid matching scheme.

- Second, we study the influence of the loss weight λ associated with the one-to-many matching loss $\mathcal{L}_{\mathrm{one2many}}$ in Table 16. We can see that our approach is not sensitive to the choice of λ and we simply choose $\lambda=1$ for all experiments by default.

- Third, to investigate why hybrid matching fails to improve the performance when choosing $K = 1$. We em-

pirically find that the bounding boxes located at the top $300 \sim 600$ are of poor localization quality, where simply replacing the top $0 \sim 300$ bounding box predictions with top $300 \sim 600$ bounding box predictions during training and evaluation causes a significant performance drop (47% vs. 42.5%). To address this problem, we study the influence of a more advanced selection scheme within one-to-many matching branch: (i) use two independent classification heads on the output of the transformer encoder to perform one-to-one/one-to-many (here we use $K + 1$ groups of ground truth in one-to-many branch) matching and select top $300/300 \times (K + 1)$ predictions respectively, (ii) use the remaining $300 \times (K)$ predictions in the one-to-many matching branch by filtering out the duplicated ones presented in the top 300 predictions of the one-to-one matching branch. Based on the above-improved selection scheme, we achieve 47.9% when choosing $K = 1$ and observe no advantages when choosing $K = 6$ compared to the original selection strategy.

- Fourth, to verify whether this observation still holds on DINO-DETR, Table 17 reports the detailed ablation results on DINO-DETR. We can see that the gains of (contrast) query denoising also mainly comes from the better optimization of the transformer encoder instead of the decoder.
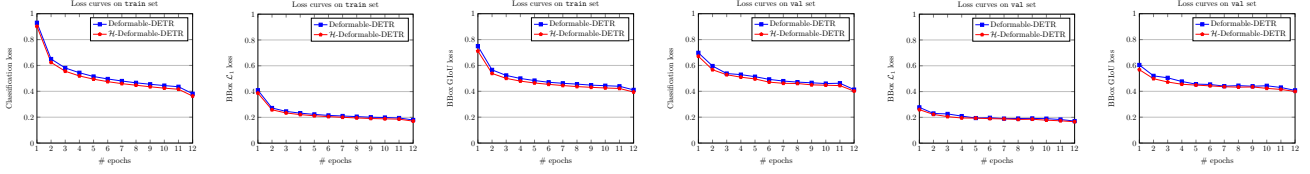
Figure 8. Illustrating the loss curves on `train` and `val`. Our approach helps the optimization of Deformable-DETR, thus decreasing the loss values on both `train` and `val` of COCO object detection benchmark.

Algorithm 1 Hybrid Matching Loss Function.

```
1   def naive_hybrid_loss(
2           outputs_one2one,
3           outputs_one2many,
4           targets,
5           k_repeated_targets):
6       loss_dict_one2one = criterion(
7           outputs_one2one, targets)
8       loss_dict_one2many = criterion(
9           outputs_one2many, k_repeated_targets)
10      def criterion(
11          prediction,
12          target):
13          cost_matrix = compute_cost()
14          indices = hungarian_matching(cost_matrix)
15          loss(prediction, target, indices)
16
17
18  def optimized_hybrid_loss(
19          outputs_one2one,
20          outputs_one2many,
21          targets,
22          k_repeated_targets):
23      cost_matrix_one2one, cost_matrix_one2many =
24      compute_cost(
25          concatenate(outputs_one2one, outputs_one2many),
26          concatenate(targets, k_repeated_targets)
27      )
28      indices_one2one = hungarian_matching(
29          cost_matrix_one2one)
30      indices_one2many = hungarian_matching(
31          cost_matrix_one2many)
32      loss(
33          concatenate(outputs_one2one, outputs_one2many),
34          concatenate(target, k_repeated_targets),
35          concatenate(indices_one2one, indices_one2many))
```

line and our hybrid matching scheme in Figure 7. It can be seen that our approach mainly improves the recall rate of the baseline method, thus also ensuring the lower false negative rates.

E. Accelerate Hybrid Matching

In the original implementation, we simply perform the one-to-one/one-to-many matching and loss computation with two independent functions, thus increasing the overall training time of each epoch from 75min to 85min when compared to the baseline that uses the same total number of queries. To decrease the additional latency, we merge the computation of their cost matrices and loss functions in Algorithm 1. Based on the accelerated implementation, we decrease the original training time from 85min to 80min.

Acknowledgement

For example, using independent transformer decoders and independent box & classification heads only suffers from $0.1\% \downarrow$ drop ($49.1\% \rightarrow 49.0\%$) while further using an independent transformer encoder contributes to another $0.5\% \downarrow$ drop ($49.1\% \rightarrow 48.5\%$).

- Fifth, we illustrate the loss curves of our $\mathcal{H}$-Deformable-DETR and the baseline Deformable-DETR in Figure 8. To ensure fairness, we only consider the $\mathcal{L}_{\text{one2one}}$ through the whole training procedure. Accordingly, we can see our approach achieves both lower training loss values and lower validation loss values, which shows that the additional one-to-many matching branch could ease the optimization of the one-to-one matching branch.

- Sixth, we compare the precision-recall curves of the base-