Improving the Efficiency-Accuracy Trade-off of DETR-Style Models in Practice

Yumin Suh[1] Dongwan Kim[2] Abhishek Aich[1] Samuel Schulter[1] Jong-Chyi-Su[1]
Bohyung Han[2] Manmohan Chandraker[1]

[1]NEC Laboratories America [2]Seoul National University

Abstract

We aim to provide a comprehensive view of the inference efficiency of DETR-style detection models. We explore the effect of basic efficiency techniques and identify the factors that are easy to implement, yet effectively improve the efficiency-accuracy trade-off. Specifically, we investigate the effect of input resolution, multi-scale feature enhancement, and backbone pre-training. Our experiments support that 1) adjusting the input resolution is a simple yet effective way to achieve a better efficiency-accuracy trade-off. 2) Multi-scale feature enhancement can be lightened with a marginal decrease in accuracy, and 3) improved backbone pre-training can further improve the trade-off.

1. Introduction

DETR (DEtection TRansformer) architectures are currently the state-of-the-art in detection (*e.g.*, 66% AP for detection in COCO test split [1]) and have been explored extensively [11, 15, 24–26]. Although these models often achieve high accuracy, their practical inference speeds are not as well-documented. This gap raises a critical question: "*How can the computational cost of DETR-style models be minimized in practice while simultaneously mitigating any potential decrease in accuracy*?"

To this end, we investigate the existing techniques to improve their inference efficiency, *i.e.*, mapping optimization (*e.g.* TensorRT) [10, 20], architecture design [12], backbone pretraining, and data augmentation to allow smaller input resolution [22]. Instead of developing a new efficiency technique for a certain scope, we aim to provide a comprehensive view of accuracy-latency trade-off obtainable from each technique and their combinations. We choose basic methods for each scope as summarized in Table 1. The key observations from our analysis are summarized below.

- Adjusting the input resolution is a simple yet effective way to achieve a better efficiency-accuracy trade-off. For example, in our experiments, larger backbone with smaller input resolution (DINO-swinB, 640×640) achieves 3.5% higher AP with fewer GFLOPs, compared to a smaller backbone with larger resolution (DINO-res50, 857×1045) on the COCO validation split.
- Multi-scale feature enhancer (or the transformer encoder) can be lightened with marginal decrease in accuracy.
- Improved backbone pre-training can further enhance the trade-off relation.

2. Scope of Analysis

There are various techniques to enhance the inference efficiency through a range of scopes [10]. We summarize the scopes of our analysis and the adopted techniques for each in Table 1. In each scope, we selected techniques that *make clear efficiency gains*, *are simple to apply*, and with preference to the methods that are *applicable in general*. In particular, we consider mapping optimizations, data precision, input resolution, backbone pre-training, and multi-scale feature enhancer. Other popular techniques, such as token sparsification, knowledge distillation, pruning, multi-task models, and neural architecture search, are out of the scope of this report. We assume GPU (NVIDIA 2080Ti) as the hardware accelerator.

Mapping optimization is a sequence of hardware instructions that defines the execution of operations on a specific hardware accelerator, including computation and memory loading [10]. Optimization of mapping is important because inference efficiency can differ a lot depending on how the operations in deep learning models are mapped to the hardware instructions. We use TensorRT 8.6.1 [2] from NVIDIA that optimizes the mapping for NVIDIA GPUs.

Data precision. Using half-precision or mixed precision in inference can reduce the computational cost of inference

[1] https://paperswithcode.com/sota/object-detection-on-coco

[2] https://github.com/NVIDIA/TensorRT

Table 1. Scope of the analysis and the selected options

Scope	Method
Mapping optimization	TensorRT
Data precision	FP16, amp
Input resolution	640×640, 768×768
Backbone	ResNet50, SwinB
Backbone pre-training	ImageNet1k, SSLD [4]
Multi-scale enhancer	LiteDETR [12]

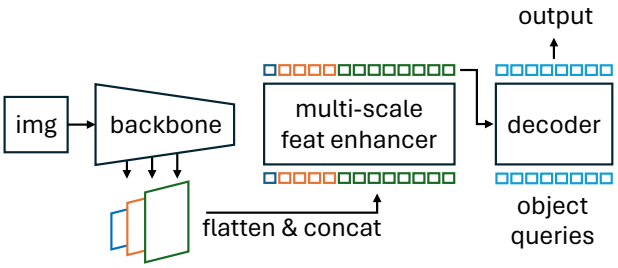

Figure 1. Overview of the DETR-style model architecture.

by a factor of 2 compared to full-precision without sacrificing model accuracy. Quantization can further reduce required memory storage, memory bandwidth, latency, and energy consumption [9, 10]. We use mixed FP16 and FP32 (amp) by default, where FP16 is used by default and FP32 is selectively adopted for modules with exponential operations that are sensitive to the precision.

Model. We focus on the DETR-style [2] architecture that consists of three modules namely, backbone, multi-scale feature enhancer (encoder), and task decoder as demonstrated in Figure 1. The backbone takes input images and outputs multi-scale feature maps. The multi-scale feature enhancer takes these multi-scale feature maps as inputs, feeds forward them through multi-scale deformable attention [25] layers to output enhanced feature maps. In the task decoder, a fixed number of object queries cross-attend to the enhanced feature maps from the multi-scale feature enhancer to predict the task outputs. We use DINO [24] architecture as our baseline.

3. Experiments and Observations

In this section, we first analyze the latency breakdowns of models of different sizes for each module. We then examine the effect of each approach for latency reduction, on both accuracy and inference efficiency. We use AP to measure accuracy, following the standard evaluation protocol [24]. We use latency with batch size 1 as a measure of inference efficiency.

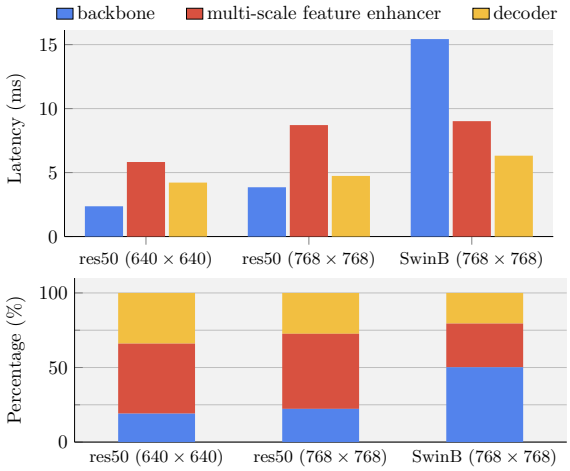

Figure 2. Latency breakdown with different backbones (ResNet50, SwinB) and input resolutions (640×640, 768×768), in latency (top) and percentage (bottom). Latency is measured using 2080Ti with a TensorRT engine with FP16.

3.1. Latency breakdown of the base model

Assuming the same architecture, computational cost of convolutional neural networks (*e.g.* ResNet50 [7]) is approximately linear to the input resolution, *i.e.* $\mathcal{O}(HW)$ [3]. Similarly, computational cost of transformers with local attention (*e.g.* Swin [17]) and multi-scale deformable attention [25] is linearly proportional to the number of input queries. Therefore, for DINO with ResNet50 and SwinB backbones, computational costs of backbone and multi-scale feature enhancers are linearly proportional to the input size, $\mathcal{O}(HW)$.

We analyze the latency of each component (backbone, encoder, and decoder) in the models with different backbones and input resolutions. Figure 2 shows the latency breakdown of DINO [24]. From the input resolution 640×640 to 768×768, latency increases approximately linear to the number of input pixels. Backbone becomes the bottleneck for models with higher accuracies, when the backbone is larger (*e.g.*, SwinB).

3.2. Effect of input resolution

Detection accuracy differs for different object sizes. As shown in Figure 3, smaller objects are generally harder to detect. On the other hand, computational cost spent in backbone, multi-scale feature enhancer is linearly proportional to the number of input pixels.

It implies that depending on the target range of object sizes, reducing the input resolution is a simple yet effective way to improve the efficiency-accuracy trade-off. For

[3] For input resolution $(H \times W)$, computational cost of each convolution layer with kernel size $K \times K$ is $\mathcal{O}(HWK^2CN)$, where C is the number of channels and N is the number of filters.

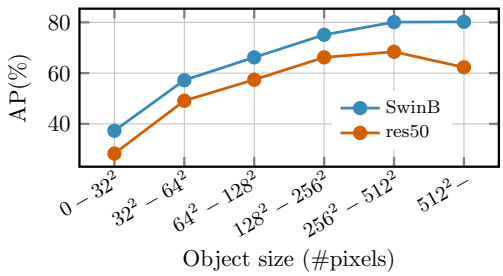

Figure 3. Detection accuracy across varying object size ranges for different backbones.

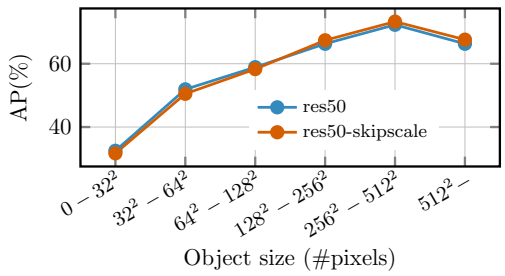

Figure 4. Detection accuracy across varying object size ranges for different multi-scale feature enhancements.

Table 2. Effect of the random zoom-out augmentation. AP and APs (%) on the COCO validation set are reported for different backbones and input resolutions, with and without random zoom-out augmentation. APs denotes the AP for small objects following the standard COCO evaluation.

Backbone	Aug	Input resolution (AP / APs)		
		640 × 640	768 × 768	Orig
res50		45.9 / 25.5	47.6 / 28.6	50.2 / 32.6
	✓	48.4 / 31.2	50.0 / 32.8	-
res50+		49.1 / 27.7	50.6 / 32.7	53.1 / 38.6
	✓	51.3 / 33.3	52.7 / 36.5	-
SwinB		53.7 / 34.5	55.2 / 37.3	57.2 / 40.5
	✓	55.1 / 36.8	56.3 / 38.8	-

Table 3. Effect of the multi-scale feature enhancement. AP and APs (%) on the COCO validation set are reported for different backbones and different input resolutions, 640 × 640 and 768 × 768. APs denotes the AP for small objects following the standard COCO evaluation. Random zoom-out augmentation is applied in all experiments.

Res	Multi-scale	Backbone (AP / APs)		
		res50	res50+	SwinB
640	skipscale	47.3 / 29.0	50.5 / 30.4	53.6 / 34.5
	orig	48.4 / 31.2	51.3 / 33.3	55.1 / 36.8
768	skipscale	49.1 / 31.9	52.0 / 35.0	55.1 / 36.6
	orig	50.0 / 32.8	52.7 / 36.5	56.3 / 38.8

example, if the application targets only large objects (*e.g.* #pixels in the bounding box $> 512^2$), reducing the input resolution of DINO-SwinB from $H \times W$ to $H/2 \times W/2$ reduces the inference cost by 75%. Since the object sizes are halved in the reduced input resolution ($> 256^2$) the decrease in accuracy is marginal at 0.1% AP (Figure 3).

It also implies that improving the detection accuracy for smaller objects while minimizing the increase in inference cost is a good strategy to achieve a better trade-off between accuracy and efficiency. This allows the model to use smaller input resolution to reduce computational cost while maintaining accuracy. One straight forward way is to augment data to include more small objects during training. To address this need, YOLOv4 [1] introduced mosaic augmentation, while some real-time detection methods [16, 18] utilized random zoom-out augmentation. These techniques aim to expose the model to smaller objects more often during training. Among them, we applied random zoom-out augmentation [16]. The black arrows in Figure 5 demonstrates the effect of smaller input resolution. It shows that larger model with smaller input resolution (SwinB, 640 × 640) achieves 5% higher AP with fewer GFLOPs, compared to a smaller backbone with larger resolution (res50, 857 × 1045) on the COCO validation set. Table 2 shows that the random zoom-out augmentation is more effective in improving the detection accuracy for the smaller objects.

3.3. Effect of multi-scale feature enhancer

Multi-scale feature enhancer is a module that takes multi-scale feature maps from a backbone and enhances their representation through within-scale and cross-scale attention. It is also called as a *neck* [1] where other instances include FPN [14], PAN [1], and Bi-FPN [23]. Most recent DETR-style models [3, 13, 24] adopt the neck architecture proposed in Deformable DETR [25], which consists of a sequence of layers composed of self-attention and multi-scale deformable attention. (*msdeformattn*). Specifically, *msdeformattn* effectively reduces the computational cost from quadratic ($O(H^2W^2)$) to linear ($O(HW)$) by attending to a fixed number of points where the locations to attend are dynamically computed depending on the input. However, it is still linearly proportional to the input resolution.

Multi-scale enhancer consists of a sequence of transformer layers, where each layer takes the concatenation of flattened feature maps from all scales and enhances them through the within-scale and cross-scale attentions. Recent research [12, 18] on detection observed that the majority of its computation could be saved with marginal or no accuracy drop, by selectively skipping the subset of scales in each layer. Among them, we adopted LiteDETR [12] without the key-aware deformable attention, for its simplic-

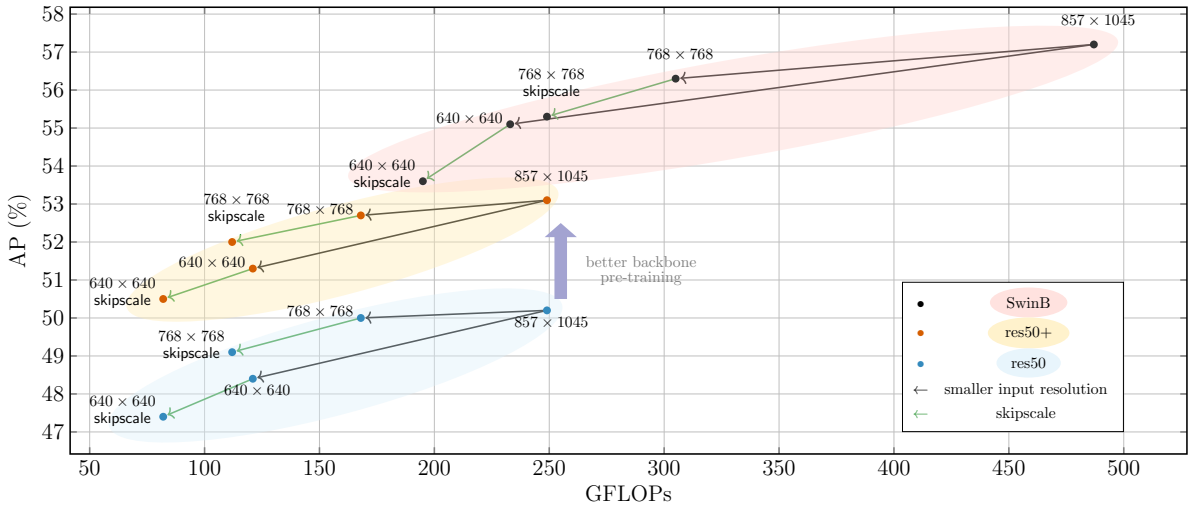

Figure 5. Accuracy-efficiency trade-off results on COCO detection validation set. Using smaller input resolution effectively reduces computational costs while preserving accuracy (black arrows). Enhanced backbone pre-training using SSLD improves accuracy given the same computational cost (purple arrow). Skipping scales in the multi-scale feature enhancer moves along the Pareto front (green arrows). GFLOPs is averaged over test images.

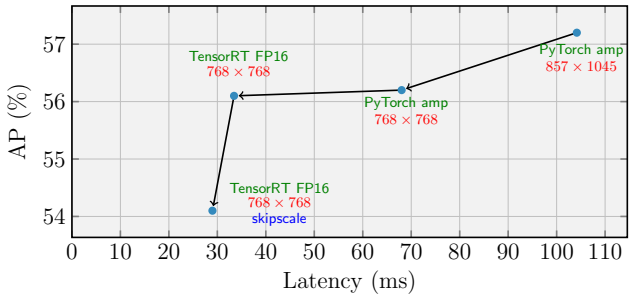

Figure 6. Effect of efficiency techniques in accuracy and latency. The smaller the slope of the arrow, the more efficient it is for the accuracy-efficiency trade-off.

ity and effectiveness. We denote it *skipscale* to distinguish it from the original LiteDETR. Green arrows in Figure 5 shows the effect of the lightened multi-scale feature enhancers. It shows that multi-scale features can be greatly lightened with marginal accuracy drop. Figure 4 and Table 3 show the effect of lightened multi-scale feature enhancement for different object sizes.

3.4. Effect of backbone pre-training

A standard approach to train detection models is to first pre-train the model using generic loss (*e.g.*, classification [5], contrastive loss [19, 21], reconstructive loss [6, 8]) and then finetune the pretrained backbone together with the task-specific modules to the target downstream task. Pre-training can learn representations that align better with the target task or robust over domains by leveraging larger amount of data over diverse domains. Since it does not affect the inference speed, it can improve the accuracy-efficiency trade-off via enhancing accuracy. We compare detection results of the same backbones with two different pre-trainings. Figure 5 shows that the backbone pre-trained with SSLD [4] (res50+) [4] achieves 4% higher AP compared to the baseline with the standard pre-training using the ImageNet-1k supervision (res50). It empirically supports that efficient backbone architecture with enhanced pre-training strategy can achieve a better accuracy-efficiency trade-off by enhancing the accuracy without affecting inference speed as shown with the purple arrow (res50 → res50+).

3.5. Results of the combined efficiency techniques

The above efficiency techniques complement each other and can be used together to further improve performance. Figure 5 shows that the effect of their combinations. The smaller the slope of the arrow, the more effective it is for the accuracy-efficiency trade-off. Figure 6 plots the efficiency in latency and shows that their relative gains are preserved.

4. Conclusion

In this report, we explored the efficiency-accuracy trade-offs of different efficiency techniques and their combinations. Our experiments support that 1) adjusting the input resolution is a simple yet effective way to achieve a better efficiency-accuracy trade-off. 2) Multi-scale feature enhancement can be lightened with a marginal decrease in accuracy, and 3) improved backbone pre-training can further improve the trade-off.

[4]SSLD [4] distills knowledge from ResNeXt101 to ResNet50 using unlabeled data collected from ImageNet22k

References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection, 2020. 3

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 2

[3] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022. 3

[4] Cheng Cui, Ruoyu Guo, Yuning Du, Dongliang He, Fu Li, Zewu Wu, Qiwen Liu, Shilei Wen, Jizhou Huang, Xiaoguang Hu, et al. Beyond self-supervision: A simple yet effective network distillation alternative to improve backbones. *arXiv preprint arXiv:2103.05959*, 2021. 2, 4

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern recognition*, pages 248–255. IEEE, 2009. 4

[6] Alaaeldin El-Nouby, Michal Klein, Shuangfei Zhai, Miguel Angel Bautista, Alexander Toshev, Vaishaal Shankar, Joshua M Susskind, and Armand Joulin. Scalable pre-training of large autoregressive image models. *arXiv preprint arXiv:2401.08541*, 2024. 4

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 2

[8] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 4

[9] Mark Horowitz. 1.1 computing's energy problem (and what we can do about it). In *2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC)*, pages 10–14. IEEE, 2014. 2

[10] Sehoon Kim, Coleman Hooper, Thanakul Wattanawong, Minwoo Kang, Ruohan Yan, Hasan Genc, Grace Dinh, Qijing Huang, Kurt Keutzer, Michael W Mahoney, et al. Full stack optimization of transformer inference: a survey. *arXiv preprint arXiv:2302.14017*, 2023. 1, 2

[11] 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. 1

[12] Feng Li, Ailing Zeng, Shilong Liu, Hao Zhang, Hongyang Li, Lei Zhang, and Lionel M Ni. Lite detr: An interleaved multi-scale encoder for efficient detr. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18558–18567, 2023. 1, 2, 3

[13] Feng Li, Hao Zhang, Huaizhe Xu, Shilong Liu, Lei Zhang, Lionel M Ni, and Heung-Yeung Shum. Mask dino: Towards a unified transformer-based framework for object detection and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3041–3050, 2023. 3

[14] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 3

[15] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. DAB-DETR: Dynamic anchor boxes are better queries for DETR. In *International Conference on Learning Representations*, 2022. 1

[16] 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. 3

[17] 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

[18] Wenyu Lv, Shangliang Xu, Yian Zhao, Guanzhong Wang, Jinman Wei, Cheng Cui, Yuning Du, Qingqing Dang, and Yi Liu. Detrs beat yolos on real-time object detection, 2023. 3

[19] Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. Scaling open-vocabulary object detection. *Advances in Neural Information Processing Systems*, 36, 2024. 4

[20] NVIDIA Corporation. TensorRT. https://developer.nvidia.com/tensorrt, Year. [Online; accessed Day-Month-Year]. 1

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 4

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

[23] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790, 2020. 3

[24] 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. In *International Conference on Learning Representations*, 2023. 1, 2, 3

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

[26] Zhuofan Zong, Guanglu Song, and Yu Liu. Detrs with collaborative hybrid assignments training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6748–6758, 2023. 1