Towards Data-Efficient Detection Transformers (Supplementary Material)

Wen Wang[1], Jing Zhang[2], Yang Cao[1,3], Yongliang Shen[4], Dacheng Tao[5,2]

[1]University of Science and Technology of China [2]The University of Sydney
[3]Institute of Artificial Intelligence, Hefei Comprehensive National Science Center
[4]Zhejiang University [5]JD Explore Academy, China
wangen@mail.ustc.edu.cn jing.zhang1@sydney.edu.au
forrest@ustc.edu.cn syl@zju.edu.cn dacheng.tao@gmail.com

A Data Efficiency

Although there is no strict definition of data efficiency, it has been studied under various contexts [16,10,15,6]. As described in Section 1 of our main text, the requirement of more training data brings two issues: (a) more human labors are needed to collect and label enough training data; (b) more computational costs are required to train the model. In this paper, we aim to alleviate both these two issues of existing detection transformers. Thus, most of our experiments are performed on small-size datasets and implemented with a short training schedule of 50 epochs.

Table S1. Performance comparison of CondDETR variants trained with a long training schedule, experimented on down-sampled COCO 2017 dataset. "Rate" indicates the sample rate. The number of training epochs is increased to ensure the computational cost is the same as that of full COCO 2017 training.

Rate	Method	Epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params	FLOPs	FPS
0.01	CondDETR [11]	5000	10.0	21.1	8.3	2.6	10.7	15.7	43M	90G	39
	DE-CondDETR	5000	13.2	23.5	12.8	5.4	14.2	17.3	44M	107G	28
	DELA-CondDETR	5000	13.4	24.1	13.0	5.6	14.4	17.7	44M	107G	28
0.02	CondDETR [11]	2500	14.7	28.6	13.3	4.3	14.8	23.7	43M	90G	39
	DE-CondDETR	2500	17.5	30.7	17.4	7.3	18.9	24.7	44M	107G	28
	DELA-CondDETR	2500	18.4	32.0	18.6	8.0	18.8	25.9	44M	107G	28
0.05	CondDETR [11]	1000	20.1	36.8	19.4	6.5	21.0	30.9	43M	90G	39
	DE-CondDETR	1000	23.3	39.3	23.7	10.5	24.9	32.4	44M	107G	28
	DELA-CondDETR	1000	23.8	40.0	24.4	10.7	25.4	33.2	44M	107G	28
0.1	CondDETR [11]	500	24.9	43.6	24.4	8.9	26.6	37.7	43M	90G	39
	DE-CondDETR	500	27.5	45.0	28.2	13.6	29.1	37.0	44M	107G	28
	DELA-CondDETR	500	28.3	46.1	29.4	14.4	30.0	38.5	44M	107G	28

In this section, we ablate the effectiveness of our method on alleviating human labor, by training on small-size datasets but with a much longer training schedule. Specifically, we use the same sub-sampled COCO 2017 dataset in Section 5 of our main text. But we increase the number of training epochs to match the computational cost of full COCO 2017 dataset training. The experiments are based on CondDETR [11] variants. All models are trained with a batch size of 32 and the learning rate is decayed after training for 0.8 times the total training epochs.

As can be seen from Table S1, with a much longer training schedule and more training costs, the performance of the CondDETR baseline is improved compared with the results under a short training schedule in Fig. 4 (b) of our manuscript, especially when the number of training images is small. However, our DE-CondDETR and DE-CondDETR still consistently outperform the CondDETR baseline by a large margin, which manifests the effectiveness of our method in alleviating human labor for data collection.

B Fine-tuning from COCO pre-trained model weights

Table S2. Training on Cityscapes with COCO pre-trained weights.

Method	Pre-training on COCO		Fine-tuning on Cityscapes			
	Epochs	AP	Epochs	AP	AP_{50}	AP_{75}
DETR	500	42.0	50	25.7	47.6	24.3
			300	29.4	51.5	27.7
DELA-DETR	N/A (Scratch)		50	24.5	46.2	22.5
	50	41.9	50	33.4	54.9	33.4
CondDETR	50	40.2	50	29.8	55.1	27.5
DELA-CondDETR	N/A (Scratch)		50	29.5	52.8	27.6
	50	43.0	50	35.1	58.6	35.2

We conduct experiments to train detection transformers from COCO pre-trained weights. As shown in Table S2, the performance of DETR and CondDETR are significantly improved, and slightly outperform our DELA models trained from scratch. However, even better results can be achieved when our DELA models are trained from their corresponding COCO pre-trained weights.

What's more interesting is that our DELA models can also benefit from COCO pre-trained DETR or CondDETR weights, thanks to our minimum modifications to the model structures. For example, DELA-CondDETR achieves 32.4 AP when fine-tuned from COCO pre-trained CondDETR, and DELA-DETR achieve 28.6 AP when fine-tuned from COCO pre-trained DETR.

C More Implementation Details

More details for experiments in Fig. 1. Experiment results on both the sample-rich COCO 2017 dataset and the small-size Cityscapes dataset are summarized in Fig. 1 of our manuscript. The results on COCO 2017 are collected from the corresponding original papers, while the results on Cityscapes are based on our re-implementations. Specifically, we follow the default training setting of corresponding methods [13,14,21,2,11,17,5]. The only difference is that we use a small batch size of 8 to guarantee enough training iterations on the small-size dataset. For SMCA [5], since only the single-scale version is made publicly available, we adopt its single-scale variant, denoted as SMCA-SS. All Cityscapes experiments are repeated for 5 runs with different random seeds and the averaged results are reported.

More details for the model transformation. Similar to other experiments on small-size datasets, we use a small batch size of 8 to guarantee enough training iterations on Cityscapes. Sparse RCNN is trained with focal loss [8] and 300 queries, under the same data augmentation pipeline as DETR.

More details for the experiments in Section 5. (a) For the ablation study on the number of multi-scale features, the 64× down-sampled feature is obtained by applying a 3×3 convolution layer with a stride of 2 on the 32× down-sampled feature, following DeformDETR [21]. (b) For the evaluation of generalization to the sample-rich dataset, we re-implement DETR and CondDETR on COCO 2017 dataset for a fair comparison with our method.

D Pseudo-code for the implementation of DE-DETRs

```
def forward(image_feats, query_feats, bbox):
    # image_feats: (B, D, H, W), where B is the batch size
    # query_feats: (B, N, D), where N is the number of queries
    # bbox: (B, N, 4), bounding box prediction made by previous decoder layer

    # Self-attention
    query_feats = self_attn(query_feats) # (B, N, D)

    # Local Feature Sampling
    sparse_feats = RoIAlign(image_feats, bbox) # (B, N, D, K, K)

    # For cross-attention, the batch size is treated as B*N for parallel decoding
    sparse_feats = sparse_feats.view(B*N, D, K*K).permute(0, 2, 1) # (B*N, K*K, D)
    query_feats = query_feats.view(B*N, 1, D)
    query_feats = cross_attn(query_feats, sparse_feats).view(B, N, D)

    # Predictions in the current decoder layer
    class_probs = classifier(query_feats)
    bbox = regressor(query_feats) # for feature sampling in the next decoder layer

    return query_feats, class_probs, bbox
```

Fig. S1. Pseudo-code for the forward function of a single decoder layer with local feature sampling. We use a single-scale image feature for illustration.

With the proposed local feature sampling in the decoder layer, each object query attend to different set of keys. To facilitate parallel decoding in the cross-attention layer, we treat different queries as individual samples in a batch.

Specifically, suppose a batch of input queries to the decoder has a shape of (B, N, D), where B is the batch size. In cross-attention, the queries are viewed as $(B \times N, 1, D)$ for parallel decoding. The pseudo-code in Fig. S1 illustrate the forward function of a single decoder layer with local feature sampling.

E Discussions on Local Feature Sampling

Our model transformation shows that sparse feature sampling from local areas is critical to data efficiency. Though PnP-DETR [17] also attempts to sample sparse features, the features are sampled from the entire image, instead of a local area in the image. As a result, the sampled features may contain multiple instances and the model still has to learn to focus on specific objects from more training data. By contrast, Sparse RCNN, DeformDETR, and our method sample sparse features from local object regions, thus alleviating the data-hungry issue.

F DeformDETR with Label Augmentation

Since DeformDETR already performs sample feature sampling from local areas and multi-scale feature fusion with the advanced Deformable Attention [21], we do not modify its model structure. Instead, we simply combine our label augmentation method with DeformDETR, which is denoted as LA-DeformDETR.

Table S3. DeformDETR trained with the proposed label augmentation (LA) on Cityscapes. "SF", "MS", and "LA" represent sparse feature sampling, multi-scale feature fusion, and label augmentation, respectively.

Method	SF	MS	LR	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params	FLOPs	FPS
DeformDETR [21]	✓	✓		27.3	49.2	26.3	8.7	28.2	45.7	40M	174G	28
LA-DeformDETR	✓	✓	✓	28.6	52.2	27.4	8.9	28.9	47.9	40M	174G	28

Experiments on Cityscapes. As can be seen from Table S3, the DeformDETR is a data-efficient model that achieves an even better performance than Sparse RCNN on Cityscapes. However, our label augmentation can further improve its data efficiency and achieve a 1.3 AP gain to the strong baseline.

Experiments on the sub-sampled COCO 2017. We also evaluate the performance of our LA-DeformDETR on the sub-sampled COCO 2017 dataset. As shown in Fig. S2, our method consistently outperforms the DeformDETR baseline under varying sampling rates.

G Ablations based on DELA-DETR

To gain a more comprehensive understanding of each component in our method, we also conduct ablation studies based on DELA-DETR, following the ablation studies in Section 5.2 of our main text.

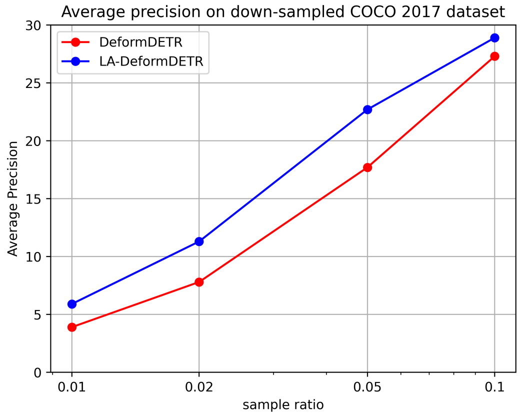

Fig. S2. Performance comparison of DeformDETR variants on sub-sampled COCO 2017 dataset. Note the sample ratio is shown on a logarithmic scale. As can be seen, the proposed label augmentation consistently improves the model performance under varying data sampling ratios.

Table S4. Ablations on each component in DELA-DETR. “SF”, “MS”, and “LA” represent sparse feature sampling, multi-scale feature fusion, and label augmentation, respectively. † indicates the query number is increased from 100 to 300.

Method	Epochs	SF	MS	LA	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params	FLOPs	FPS
DETR [2]	300				11.5	26.7	8.6	2.5	9.5	25.1	41M	86G	44
LA-DETR	300			✓	16.8	36.9	13.3	3.0	13.4	35.5	41M	86G	44
	50	✓			16.3	34.9	12.9	2.3	12.0	35.9	42M	85G	36
DE-DETR	50	✓	✓		21.7	41.7	19.2	4.9	20.0	39.9	42M	88G	34
DELA-DETR	50	✓	✓	✓	20.6	40.1	18.4	4.6	18.9	37.5	42M	91G	29
DE-DETR†	50	✓	✓		22.4	41.3	20.9	6.0	21.3	39.7	42M	91G	29
DELA-DETR†	50	✓	✓	✓	24.5	46.2	22.5	6.1	23.3	43.9	42M	91G	29

Effectiveness of each module. As shown in Table S4, both sparse feature sampling from local areas and multi-scale feature significantly improve the data efficiency of DETR, bringing a 4.8 and 5.4 gain on AP even under a much shorter training schedule. Besides, we also apply label augmentation to DE-DETTR. However, directly applying label augmentation makes the performance worse. We conjecture that with only 100 queries, the positive sample ratio becomes too high for training, particularly for the Cityscapes dataset, where many images contain dense object scenes. To solve this problem, we improve the number of queries from 100 to 300, as did in CondDETR and DeformDETR. As can be seen, DELA-DETR† outperforms DE-DETR by 2.8 AP. Moreover, from the comparison between DE-DETR† and DELA-DETR†, it can be seen that the performance gain mainly comes from our label augmentation, instead of more queries.

Table S5. Ablations on multi-scale feature levels and feature resolutions for RoIAlign, experimented based on DE-DETR. Note label augmentation is not utilized for clarity.

MS Lvls	RoI Res.	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params	FLOPs	FPS
1	1	12.5	30.8	8.1	1.8	8.9	26.9	42M	85G	36
1	4	16.3	34.9	12.9	2.3	12.0	35.9	42M	85G	36
1	7	16.5	35.6	13.0	2.3	12.5	36.5	42M	86G	36
3	4	21.7	41.7	19.2	4.9	20.0	39.9	42M	88G	34
4	4	21.1	40.8	18.9	4.3	18.6	39.5	47M	89G	33

Resolution for RoIAlign and number of multi-scale features. As can be seen from Table S5, RoIAlign with a feature resolution of 4 is both efficient and effective. And the three feature levels for multi-scale fusion achieve the optimal performance. Thus, the hyper-parameter settings of DE-DETR are exacted the same as that of DE-CondDETR.

Table S6. Ablation on the proposed label augmentation, experimented based on DE-DETR. Params, FLOPs, and FPS are omitted since they are consistent for all label augmentation settings.

Fix Time	Fix Ratio	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
–	–	21.7	41.7	19.2	4.9	20.0	39.9
2	–	24.5	46.2	22.5	6.1	23.3	43.9
3	–	23.7	43.8	21.8	5.7	21.7	43.5
4	–	22.8	42.7	21.1	5.7	21.3	41.5
5	–	22.6	43.1	20.1	5.5	20.7	40.9
–	0.1	23.8	44.8	21.1	5.6	22.4	42.0
–	0.2	23.4	43.7	21.2	6.0	21.7	41.8
–	0.25	23.1	43.4	21.0	5.4	22.1	41.1
–	0.3	23.1	43.3	21.1	5.7	21.8	41.0
–	0.4	21.9	40.4	20.5	5.1	21.0	39.8

Ablations on label augmentation. We also ablate the proposed label augmentation method with the fixed repeat time strategy and the fixed positive sample ratio strategy. As can be seen from Table S6, a fixed repeat time of 2 consistently achieve the best performance.

H Ablations on NMS

Since the proposed label augmentation performs one-to-many matching between ground truths and predictions, a duplicate remove process is required. Although more advanced duplicate removal methods, like Soft-NMS [1], can be used, we simply adopt the vanilla NMS.

NMS with different IoU thresholds. We first ablate NMS with different IoU thresholds. As can be seen from Table S7, DELA-CondDETR with different IoU

Table S7. Ablations on NMS, experiments based on DELA-CondDETR. N_t indicates the IoU threshold for NMS.

Method	N_t	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
DE-CondDETR	–	26.8	47.8	25.4	6.8	25.6	46.6
	0.7	27.1	49.2	25.3	6.8	25.9	47.5
DELA-CondDETR	0.3	28.6	51.9	26.6	7.1	27.3	49.3
	0.4	29.0	52.7	26.9	7.2	27.6	49.7
	0.5	29.3	53.2	27.1	7.3	28.0	50.0
	0.6	29.5	53.2	27.4	7.5	28.2	50.1
	0.7	29.5	52.8	27.6	7.5	28.2	50.1
	0.8	29.3	51.8	27.8	7.4	28.0	49.8
	0.9	28.5	49.6	27.5	7.1	26.9	48.7

thresholds for NMS consistently outperforms the DE-CondDETR. Moreover, NMS with a wide range of IoU thresholds from 0.4 to 0.8 can achieve good performance. We conjecture that it is easier to remove the duplicates when each positive sample is repeated only two times.

DE-CondDETR with NMS. As can be seen from Table S7, applying NMS on DE-CondDETR does not make much difference on model performance, since it follows a one-to-one matching scheme during training. This validates the performance gain of DELA-CondDETR comes from a richer supervision signal, instead of the duplicate removal process.

I Ablations on the Box Refinement

Table S8. Ablations on the single-scale sparse feature sampling, experimented based on CondDETR. RoI and Refine indicate RoIALign and cascaded bounding box refinement, respectively. All models are trained for 50 epochs.

Method	Refine	RoI	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params	FLOPs	FPS
CondDETR [11]			12.1	28.0	9.1	2.2	9.8	27.0	43M	90G	39
	✓		12.1	30.2	7.8	2.1	10.4	25.7	44M	90G	39
		✓	16.3	32.5	14.3	3.3	15.0	33.6	43M	95G	32
	✓	✓	20.4	40.7	17.7	2.9	16.9	42.0	44M	95G	32

For sparse feature sampling from local areas, both RoIAlign and cascaded bounding box refinement are included. In this section, we ablate the effectiveness of these two components. As shown in Table S8, RoIAlign alone can bring a 4.2 gain on AP. By contrast, applying the bound box refinement alone does not improve the model performance. However, when combined with RoIAlign, the bound box refinement can improve the model performance by 4.1 AP. Since the regression targets for each decoder layer are determined by the bounding box predictions made by the previous layer, we conjecture that it is important that the current decoder layer is aware of the updated reference boxes, in order to make the box refinement effective. With RoIAlign, the current decoder layer can infer the updated reference boxes from the sampled features, which have been added with the sine positional embedding. By contrast, when applying the bounding boxes refinement alone, the updated reference boxes can hardly be inferred from the global feature map of the entire image.

J Experiments on Pascal VOC

Table S9. Performance of our data-efficient detection transformers on Pascal VOC. All models are trained on trainval07+12 for 50 epochs, and evaluated on test2007.

Method	Epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params	FLOPs	FPS
DETR [2]	50	38.3	62.1	40.3	2.1	12.9	48.3	41M	86G	43
DE-DETR	50	54.8	78.7	60.7	20.2	39.2	61.6	43M	88G	33
DELA-DETR†	50	57.0	82.0	63.0	24.5	42.1	63.8	43M	91G	29
CondDETR [11]	50	55.6	82.0	60.9	15.1	34.7	63.9	43M	90G	39
DE-CondDETR	50	56.4	80.2	63.2	22.2	40.7	62.8	44M	107G	28
DELA-CondDETR	50	59.5	84.4	66.3	29.7	43.6	65.6	44M	107G	28

In this section, we also conducted experiments on based on the Pascal VOC dataset [4]. Follow the common practice, we train models on the trainval07+12 split with contains about 16.5k images and evaluated them on the test2007 split. All models are trained with a batch size of 32. The results are shown in Table S9, as can be seen, both our DE- and DELA- model variants consistently outperform the corresponding baselines.

K Visualizations

Visualization of training curves. The comparison of DETR variants is shown in Fig. S3. As can be seen, both DE-DETR and DELA-DETR† can achieve better performance and faster convergence compared with the DETR baseline. Similarly, the comparison of ConDETR variants in Fig. S4 also validates the effectiveness of our method.

Visualization demo results. We also provide demo detection results of different methods on the Cityscapes dataset, as shown in Fig. S5 and Fig. S6. It can be seen that our method can detect the distant instances overlooked by the baseline methods. Moreover, they often avoid the false positive prediction made by the baseline methods, as shown in the second rows in both Fig. S5 and Fig. S6.

L Discussions on limitations

Though effective, our method may have certain potential disadvantages: (1) A small and fixed resolution for feature sampling may be detrimental to the detec-

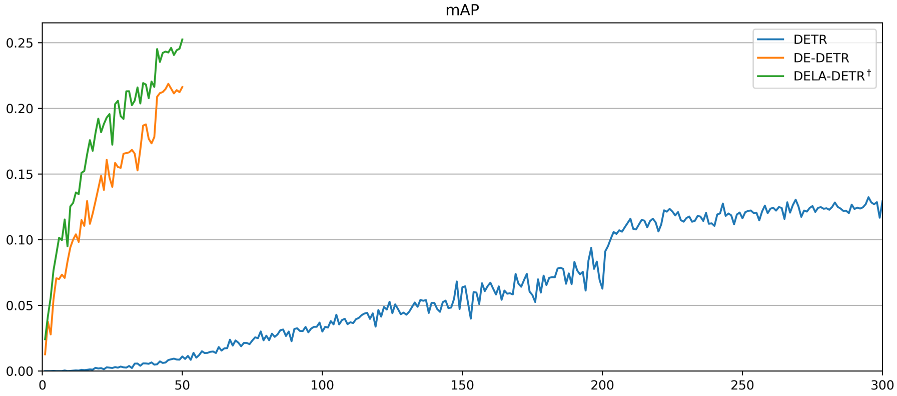

Fig. S3. Convergence curves of DETR variants on Cityscapes.

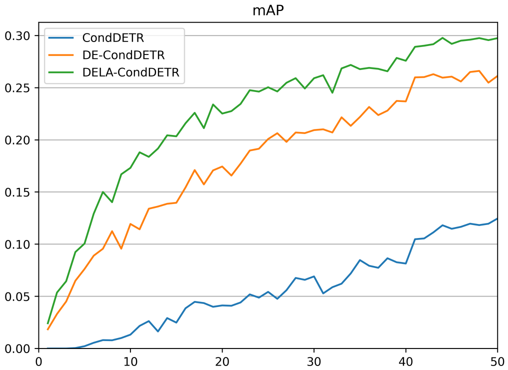

Fig. S4. Convergence curves of CondDETR variants on Cityscapes.

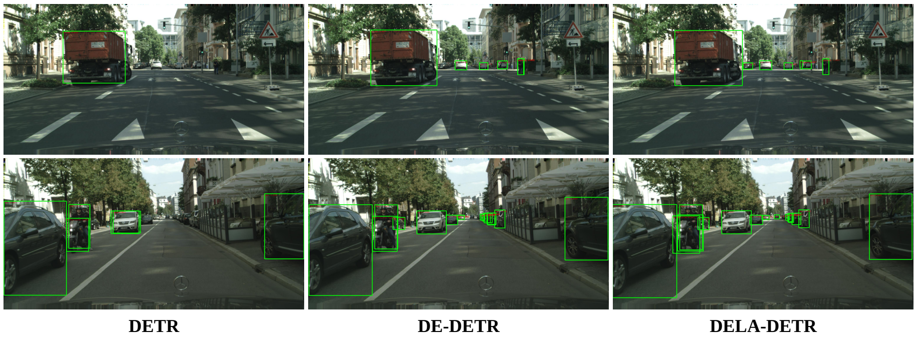

Fig. S5. Demo detection results of DETR variants on Cityscapes.

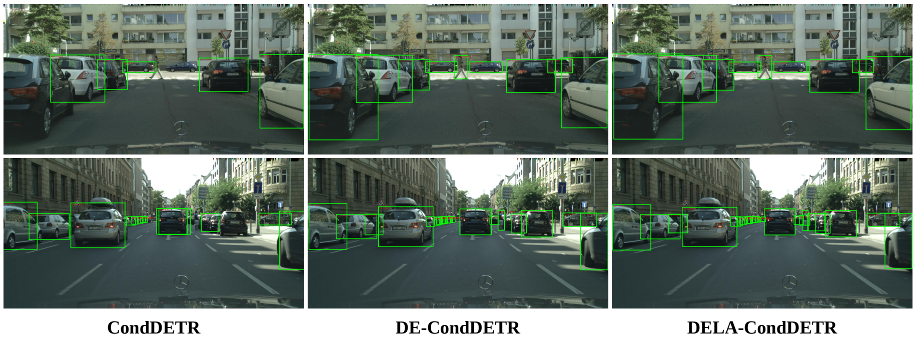

Fig. S6. Demo detection results of CondDETR variants on Cityscapes.

tion of large objects, as can be seen from the comparison between CondDETR and DELA-CondDETR in Table 7. This can be alleviated by more effective techniques that adaptively sample more feature points for large objects. (2) The sampling process may slow down the inference speed, which can be alleviated by a joint CUDA implementation of both the feature sampling and the following cross-attention processes.

M Future Work

Data efficiency of transformers on different vision tasks. With limited inductive bias, the vision transformers are often data-hungry [3]. Although the data-hungry issue of vision transformer for image classification has been studied [16], it has not been explored for other vision tasks. In this paper, we take the first step to delve into the data-hungry issue of detection transformers. As transformers become increasingly popular for vision tasks, like semantic segmentation [20,19], 3D object detection [12], and video instance segmentation [18], we hope our work will inspire the community to explore the data efficiency of transformers for different tasks.

Removing NMS. The one-to-many matching between ground-truths and predictions in our label augmentation can provide richer training supervision to alleviate the data-hungry issue. However, it also brings the need for the duplicate removal process. Though the NMS has been proved effective, we hope to remove this post-processing step to maintain the end-to-end property of detection transformers. To achieve this, we provide three possible solutions as follows. Firstly, a lightweight duplicate removal network can be trained along with the model, as did in Relation Network [7]. Secondly, an additional rank loss [9] can be applied to regularize the score of the predictions, so that the predictions matched to the original ground-truths can rank higher. In this way, the duplicate removal process is no longer needed. Thirdly, we can apply the label augmentation only on the shallower decoder layers while keeping the supervision on the deep decoder layers unchanged. In this way, the queries at the shallower decoder layers can receive a rich supervision signal, while the deeper decoder layers only select the most promising query for each target instance.

Further improving the data efficiency of existing detection transformers. Although outperforming all existing detection transformers, there is still a gap between the data efficiency of our method and the seminal Faster-RCNN-FPN. To bridge this gap, a possible solution is to gradually transform a Faster-RCNN-FPN to a Sparse RCNN, to find the key reasons for Faster-RCNN-FPN's data efficiency. Afterward, we can adjust the detection transformer structures accordingly to improve their data efficiency.

References

1. Bodla, N., Singh, B., Chellappa, R., Davis, L.S.: Soft-nms–improving object detection with one line of code. In: Proceedings of the IEEE international conference on computer vision. pp. 5561–5569 (2017)
2. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European Conference on Computer Vision. pp. 213–229. Springer (2020)
3. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
4. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International journal of computer vision **88**(2), 303–338 (2010)
5. Gao, P., Zheng, M., Wang, X., Dai, J., Li, H.: Fast convergence of detr with spatially modulated co-attention. In: Proceedings of the IEEE international conference on computer vision (2021)
6. Henaff, O.: Data-efficient image recognition with contrastive predictive coding. In: International Conference on Machine Learning. pp. 4182–4192. PMLR (2020)
7. Hu, H., Gu, J., Zhang, Z., Dai, J., Wei, Y.: Relation networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3588–3597 (2018)
8. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)
9. Liu, J., Li, D., Zheng, R., Tian, L., Shan, Y.: Rankdetnet: Delving into ranking constraints for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 264–273 (2021)
10. Liu, Y., Sangineto, E., Bi, W., Sebe, N., Lepri, B., Nadai, M.: Efficient training of visual transformers with small datasets. Advances in Neural Information Processing Systems **34** (2021)
11. Meng, D., Chen, X., Fan, Z., Zeng, G., Li, H., Yuan, Y., Sun, L., Wang, J.: Conditional detr for fast training convergence. In: Proceedings of the IEEE international conference on computer vision (2021)
12. Misra, I., Girdhar, R., Joulin, A.: An end-to-end transformer model for 3d object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2906–2917 (2021)
13. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: towards real-time object detection with region proposal networks. IEEE transactions on pattern analysis and machine intelligence **39**(6), 1137–1149 (2016)
14. Sun, P., Zhang, R., Jiang, Y., Kong, T., Xu, C., Zhan, W., Tomizuka, M., Li, L., Yuan, Z., Wang, C., 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. pp. 14454–14463 (2021)
15. Thomas, P., Brunskill, E.: Data-efficient off-policy policy evaluation for reinforcement learning. In: International Conference on Machine Learning. pp. 2139–2148. PMLR (2016)
16. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. In: International Conference on Machine Learning. pp. 10347–10357. PMLR (2021)

17. Wang, T., Yuan, L., Chen, Y., Feng, J., Yan, S.: Pnp-detr: towards efficient visual analysis with transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2021)
18. Wang, Y., Xu, Z., Wang, X., Shen, C., Cheng, B., Shen, H., Xia, H.: End-to-end video instance segmentation with transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8741–8750 (2021)
19. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: Segformer: Simple and efficient design for semantic segmentation with transformers. Advances in Neural Information Processing Systems **34** (2021)
20. Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P.H., et al.: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 6881–6890 (2021)
21. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. In: International Conference on Learning and Representations (2020)