Towards Data-Efficient Detection Transformers

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

Abstract. Detection transformers have achieved competitive performance on the sample-rich COCO dataset. However, we show most of them suffer from significant performance drops on small-size datasets, like Cityscapes. In other words, the detection transformers are generally data-hungry. To tackle this problem, we empirically analyze the factors that affect data efficiency, through a step-by-step transition from a data-efficient RCNN variant to the representative DETR. The empirical results suggest that sparse feature sampling from local image areas holds the key. Based on this observation, we alleviate the data-hungry issue of existing detection transformers by simply alternating how key and value sequences are constructed in the cross-attention layer, with minimum modifications to the original models. Besides, we introduce a simple yet effective label augmentation method to provide richer supervision and improve data efficiency. Experiments show that our method can be readily applied to different detection transformers and improve their performance on both small-size and sample-rich datasets. Code will be made publicly available at https://github.com/encounter1997/DE-DETRs.

Keywords: Data Efficiency, Detection Transformer, Sparse Feature, Rich Supervision, Label Augmentation

1 Introduction

Object detection is a long-standing topic in computer vision. Recently, a new family of object detectors, named detection transformers, has drawn increasing attention due to their simplicity and promising performance. The pioneer work of this class of methods is DETR [3], which views object detection as a direct set prediction problem and applies a transformer to translate the object queries to the target objects. It achieves better performance than the seminal Faster RCNN [31] on the commonly used COCO dataset [24], but its convergence is significantly slower than that of CNN-based detectors. For this reason, most

* This work was done during Wen Wang's internship at JD Explore Academy.
† Co-first author.
‡ Corresponding author.

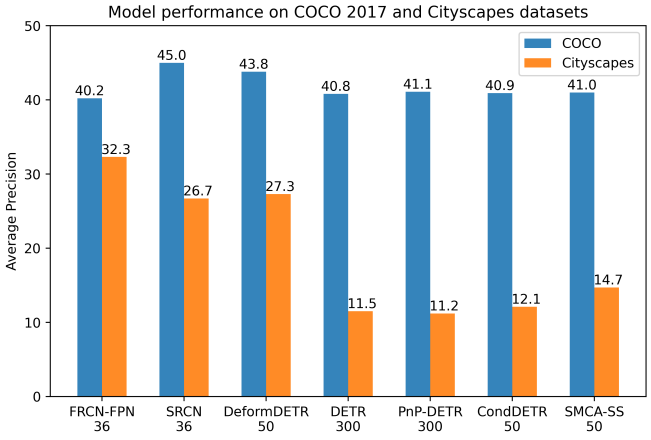

Fig. 1. Performance of different object detectors on COCO 2017 with 118K training data and Cityscapes with 3K training data. The respective training epochs are shown below the name of each method. While the RCNN family show consistently high average precision, the detection transformer family degrades significantly on the small-size dataset. FRCN-FPN, SRCN, and SMCA-SS represent Faster-RCNN-FPN, Sparse RCNN, and single-scale SMCA, respectively.

of the subsequent works have been devoted to improving the convergence of DETR, through efficient attention mechanism [50], conditional spatial query [29], regression-aware co-attention [14], *etc*. These methods are able to achieve better performance than Faster RCNN with comparable training costs on the COCO dataset, demonstrating the superiority of detection transformers.

Current works seem to suggest that detection transformers are superior to the CNN-based object detector, like Faster RCNN, in both simplicity and model performance. However, we find that detection transformers show superior performance only on datasets with rich training data like COCO 2017 (118K training images), while the performance of most detection transformers drops significantly when the amount of training data is small. For example, on the commonly used autonomous driving dataset Cityscapes [7] (3K training images), the average precisions (AP) of most of the detection transformers are less than half of Faster RCNN AP performance, as shown in Fig. 1. Moreover, although the performance gaps between different detection transformers on the COCO dataset are less than 3 AP, a significant difference of more than 15 AP exists on the small-size Cityscapes dataset.

These findings suggest that detection transformers are generally more data-hungry than CNN-based object detectors. However, the acquisition of labeled data is time-consuming and labor-intensive, especially for the object detection task, which requires both categorization and localization of multiple objects in a single image. What's more, the large amount of training data means more training iterations to traverse the dataset, and thus more computational resources are

consumed to train the detection transformers, increasing the carbon footprint. In a word, it takes a lot of human labor and computational resources to meet the training requirements of existing detection transformers.

To address these issues, we first empirically analyze the key factors affecting the data efficiency of detection transformers through a step-by-step transformation from the data-efficient Sparse RCNN to the representative DETR. Our investigation and analysis show that sparse feature sampling from local area holds the key: on the one hand, it alleviates the difficulty of learning to focus on specific objects, and on the other hand, it avoids the quadratic complexity of modeling image features and makes it possible to utilize multi-scale features, which has been proved critical for the object detection task.

Based on these observations, we improve the data efficiency of existing detection transformers by simply alternating how the key and value are constructed in the transformer decoder. Specifically, we perform sparse sampling features on key and value features sent to the cross-attention layer under the guidance of the bounding boxes predicted by the previous decoder layer, with minimum modifications to the original model, and without any specialized module. In addition, we mitigate the data-hungry problem by providing richer supervisory signals to detection transformers. To this end, we propose a label augmentation method to repeat the labels of foreground objects during label assignment, which is both effective and easy to implement. Our method can be applied to different detection transformers to improve their data efficiency. Interestingly, it also brings performance gain on the COCO dataset with a sufficient amount of data.

To summarize, our contributions are listed as follows.

- We identify the data-efficiency problem of detection transformers. Though they achieve excellent performance on the COCO dataset, they generally suffer from significant performance degradation on small-size datasets.
- We empirically analyze the key factor that affects detection transformers' data efficiency through a step-by-step model transformation from Sparse RCNN to DETR, and find that sparse feature sampling from local areas holds the key to data efficiency.
- With minimum modifications, we significantly improve the data efficiency of existing detection transformers by simply alternating how key and value sequences are constructed in the cross-attention layer.
- We propose a simple yet effective label augmentation strategy to provide richer supervision and improve the data efficiency. It can be combined with different methods to achieve performance gains on different datasets.

2 Related Work

2.1 Object Detection

Object detection [13,16,31,30,26,23,35] is essential to many real-world applications, like autonomous driving, defect detection, and remote sensing. Representative deep-learning-based object detection methods can be roughly categorized

into two-stage detectors like Faster RCNN [31] and one-stage object detectors like YOLO [30] and RetinaNet [23]. While effective, these methods generally rely on many heuristics like anchor generation and rule-based label assignments.

Recently, DETR [3] provides a simple and clean pipeline for object detection. It formulates object detection as a set prediction task, and applies a transformer [37] to translate sparse object candidates [33] to the target objects. The success of DETR has sparked the recent surge of detection transformers [50,8,14,29,25,39,12,40,4,44] and most of the following-up works focus on alleviating the slow convergence problem of DETR. For example, DeformDETR [50] propose the deformable attention mechanism for learnable sparse feature sampling and aggregates multi-scale features to accelerate model convergence and improve model performance. CondDETR [29] proposes to learn a conditional spatial query from the decoder embedding, which helps the model quickly learn to localize the four extremities for detection.

These works achieve better performance than Faster RCNN on the COCO dataset [24] with comparable training costs. It seems that detection transformers have surpassed the seminal Faster RCNN in both simplicity and superior performance. But we show that detection transformers are generally more data-hungry and perform much worse than Faster RCNN on small-size datasets.

2.2 Label Assignment

Label assignment [38,43,48,49,15,32] is a crucial component in object detection. It matches the ground truth of an object with a specific prediction from the model, and thereby provides the supervision signal for training. Prior to DETR, most object detectors [31,30,23] adopt the one-to-many matching strategy, which assigns each ground truth to multiple predictions based on local spatial relationships. By contrast, DETR makes one-to-one matching between ground truths and predictions by minimizing a global matching loss. This label assignment approach has been followed by various subsequent variants of the detection transformer [50,29,8,12,40]. Despite the merits of avoiding the duplicates removal process, only a small number of object candidates are supervised by the object labels in each iteration. As a result, the model has to obtain enough supervised signals from a larger amount of data or more training epochs.

2.3 Data-Efficiency of Vision Transformers

Vision Transformers [10,45,17,28,41,42,46,11,6] (ViTs) are emerging as an alternative to CNN for feature extractors and visual recognition. Despite the superior performance, they are generally more data-hungry than their CNN counterparts. To tackle this problem, DeiT [36] improves its data efficiency by knowledge distillation from pre-trained CNNs, coupled with a better training recipe. Liu *et al.* propose a dense relative localization loss to improve ViTs' data efficiency [27]. Unlike the prior works [36,27,2] that focus on the data efficiency issue of transformer backbones on image classification tasks, we tackle the data efficiency issue of detection transformers on the object detection task.

Model	Added	Removed	50E AP	300E AP	Params	FLOPs
SRCN	–		29.4	35.9	106M	631G
Net1	DETR Recipe	SRCN Recipe	30.6 (+1.2)	34.4 (-1.5)	106M	294G
Net2	–	FPN	23.3 (-7.3)	26.6 (-7.8)	103M	244G
Net3	transformer encoder	–	21.0 (-2.3)	27.5 (+0.9)	111M	253G
Net4	cross-attn in decoder	dynamic conv	18.1 (-2.9)	25.4 (-2.1)	42M	86G
Net5	dropout in decoder	–	16.7 (-1.4)	26.1 (+0.7)	42M	86G
Net6	–	bbox refinement	15.0 (-1.7)	22.7 (-3.4)	41M	86G
Net7	–	RoIAlign	6.6 (-8.4)	17.7 (-5.0)	41M	86G
DETR	–	initial proposals	1.6 (-5.0)	11.5 (-6.2)	41M	86G

Table 1. Model transformation from Sparse RCNN (SRCN for short) to DETR, experimented on Ciytscapes [7]. "50E AP" and "300E AP" indicate average precision after training for 50 and 300 epochs respectively. The change in AP is shown in the brackets, where red indicates drops and blue indicates gains on AP.

3 Difference Analysis of RCNNs and DETRs

As can be seen in Fig. 1, detection transformers are generally more data-hungry than RCNNs. To find out the key factors to data efficiency, we transform a data-efficient RCNN step-by-step into a data-hungry detection transformer to ablate the effects of different designs. Similar research approach has also been adopted by ATSS [47] and Visformer [5], but for different research purposes.

3.1 Detector Selection

To obtain insightful results from the model transformation, we need to choose the appropriate detectors to conduct the experiments. To this end, we choose Sparse RCNN and DETR for the following reasons. Firstly, they are representative detectors from the RCNN and detection transformer families, respectively. The observations and conclusions drawn from the transformation between them shall also be helpful to other detectors. Secondly, there is large difference between the two detectors in data efficiency, as shown in Fig. 1. Thirdly, they share many similarities in label assignment, loss design, and optimization, which helps us eliminate the less significant factors while focus more on the core differences.

3.2 Transformation from Sparse RCNN to DETR

During the model transformation, we consider two training schedules that are frequently used in detection transformers. The first is training for 50 epochs and learning rate decays after 40 epochs, denoted as 50E. And the second is training for 300 epochs and learning rate decays after 200 epochs. The transformation process is summarized in Table 1.

Alternating training recipe. Though Sparse RCNN and DETR share many similarities, there are still slight differences in their training Recipes, including the classification loss, the number of object queries, learning rate, and gradient

clip. We first eliminate these differences by replacing the Sparse RCNN training recipe with the DETR training recipe. Eliminating the differences in training recipes helps us focus more on the key factors that affect the data-efficiency.

Removing FPN. Multi-scale feature fusion has been proved effective for object detection [22]. The attention mechanism has a quadratic complexity with respect to the image scale, making the modeling of multi-scale features in DETR non-trivial. Thus DETR only takes 32× down-sampled single-scale feature for prediction. In this stage, we remove the FPN neck and send only the 32× down-sampled feature to the detection head, which is consistent with DETR. As expected, without multi-scale modeling, the model performance degrades significantly by 7.3 AP under the 50E schedule, as shown in Table 1.

Introducing transformer encoder. In DETR, the transformer encoder can be regarded as the neck in the detector, which is used to enhance the features extracted by the backbone. After removing the FPN neck, we add the transformer encoder neck to the model. It can be seen that the AP result decreases at 50E schedule while improves at 300E schedule. We conjecture that similar to ViT [10], the attention mechanism in the encoder requires longer training epochs to converge and manifest its advantages, due to the lack of inductive biases.

Replacing dynamic convolutions with cross-attention. A very interesting design in Sparse RCNN is the dynamic convolution [20,34] in the decoder, which acts very similar to the role of cross-attention in DETR. Specifically, they both adaptively aggregate the context from the image features to the object candidates based on their similarity. In this step, we replace the dynamic convolution with the cross-attention layer with learnable query positional embedding, and the corresponding results are shown in Table 1. Counter-intuitively, a larger number of learnable parameters does not necessarily make the model more data-hungry. In fact, the dynamic convolutions with about 70M parameters can exhibit better data efficiency than the parameter-efficient cross-attention layer.

Aligning dropout settings in the decoder. A slight difference between Sparse RCNN and DETR is the use of dropout layers in self-attention and FFN layers in the decoder. In this stage, we eliminate the interference of these factors.

Removing cascaded bounding box refinement. Sparse RCNN follows the cascaded bounding box regression in Cascade RCNN [1], where each decoder layer iteratively refines the bounding box predictions made by the previous layer. We remove it in this stage and as expected, the model performance degrades to some extent.

Removing RoIAlign. Sparse RCNN, like other detectors in the RCNNs family, samples features from local regions of interest, and then makes predictions based on the sampled sparse features [33]. By contrast, each content query in DETR aggregates object-specific information directly from the global features map. In this step, we remove the RoIAlign [18] operation in Sparse RCNN, with the box target transformation [16]. It can be seen that significant degradation of the model performance occurs, especially under the 50E schedule, the model performance decreases by 8.4 AP. We conjecture that learning to focus on local

object regions from the entire feature map is non-trivial. The model requires more data and training epochs to capture the locality properties.

Removing initial proposals. Finally, DETR directly predicts the target bounding boxes, while RCNNs make predictions relative to some initial guesses. In this step, we eliminate this difference by removing the initial proposal. Unexpectedly, this results in a significant decrease in model performance. We suspect that the initial proposal works as a spatial prior that helps the model to focus on object regions, thus reducing the need to learn locality from large training data.

3.3 Summary

By far, we have completed the model transformation from Sparse RCNN to DETR. From Table 1 and our analysis in Section 3.2, it can be seen that three factors result in more than 5 AP performance changes, and are key to data-efficient: (a) sparse feature sampling from local regions, *e.g.*, using RoIAlign; (b) multi-scale features which depend on sparse feature sampling to be computationally feasible; (c) prediction relative to initial spatial priors. Among them, (a) and (c) help the model to focus on local object regions and alleviate the requirement of learning locality from a large amount of data, while (b) facilitates a more comprehensive utilization and enhancement of the image features, though it also relies on sparse features.

It is worth mentioning that DeformDETR [50] is a special case in the detection transformer family, which shows comparable data efficiency to Sparse RCNN. Our conclusions drawn from the Sparse RCNN to DETR model transformation can also explain DeformDETR's data efficiency. Specifically, multi-scale deformable attention samples sparse features from local regions of the image and utilizes multi-scale features. The prediction of the model is relative to the initial reference points. Thus, all three key factors are satisfied in DeformDETR, though it was not intended to be data-efficient on small-size datasets.

4 Method

In this section, we aim to improve the data efficiency of existing detection transformers, while making minimum modifications to their original designs. Firstly, we provide a brief revisiting of existing detection transformers. Subsequently, based on experiments and analysis in the previous section, we make minor modifications to the existing data-hungry detection transformer models, like DETR [3] and CondDETR [29], to significantly improve their data efficiency. Finally, we propose a simple yet effective label augmentation method to provide richer supervised signals to detection transformers to further improve their data efficiency.

4.1 A Revisit of Detection Transformers

Model Structure. Detection transformers generally consist of a backbone, a transformer encoder, a transformer decoder, and the prediction heads. The backbone first extracts multi-scale features from the input image, denoted as $\{f^l\}_{l=1}^{L}$,

where $f^l \in \mathbb{R}^{H^l \times W^l \times C^l}$. Subsequently, the last feature level with the lowest resolution is flattened and embedded to obtain $z^L \in \mathbb{R}^{S^L \times D}$ where $S^L = H^L \times W^L$ is sequence length and D is the feature dimension. Correspondingly, the positional embedding is denoted as $p^L \in \mathbb{R}^{S^L \times D}$. Afterward, The single-scale sequence feature is encoded by the transformer encoder to obtain $z_e^L \in \mathbb{R}^{S^L \times D}$.

The decoder consists of a stack of L_d decoder layers, and the query content embedding is initialized as $\mathbf{q}_0 \in \mathbb{R}^{N \times D}$, where N is the number of queries. Each decoder layer DecoderLayer_ℓ takes the previous decoder layer's output $\mathbf{q}_{\ell-1}$, the query positional embedding p_q, the image sequence feature $\mathbf{z}_\ell$ and its position embedding p_ℓ as inputs, and outputs the decoded sequence features.

$$\mathbf{q}_\ell = \text{DecoderLayer}_\ell \left(\mathbf{q}_{\ell-1}, p_q, \mathbf{z}_\ell, p_\ell\right), \quad \ell = 1 \ldots L_\text{d}. \tag{1}$$

In most detection transformers, like DETR and CondDETR, single-scale image feature is utilized for decoder, and thus $\mathbf{z}_\ell = z_e^L$ and $p_\ell = p^L$, where $\ell = 1 \ldots L_\text{d}$.

Label Assignment. Detection transformers view the object detection task as a set prediction problem and perform deep supervision [21] on predictions made by each decoder layer. Specifically, the labels set can be denoted as $y = \{y_1, \ldots, y_M, \varnothing, \ldots, \varnothing\}$, where M denotes the number of foreground objects in the image and the $\varnothing$ (no object) pads the label set to a length of N. Correspondingly, the output of each decoder layer can be written as $\hat{y} = \{\hat{y}_i\}_{i=1}^N$. During label assignment, detection transformers search for a permutation $\tau \in T_N$ with the minimum matching cost:

$$\hat{\tau} = \underset{\tau \in T_N}{\arg\min} \sum_i^N \mathcal{L}_\text{match} \left(y_i, \hat{y}_{\tau(i)}\right), \tag{2}$$

where $\mathcal{L}_\text{match} \left(y_i, \hat{y}_{\tau(i)}\right)$ is the pair-wise loss between ground truth and the prediction with index $\tau(i)$.

4.2 Model Improvement

In this section, we make slight adjustments to data-hungry detection transformers such as DETR and CondDETR, to largely boost their data efficiency.

Sparse Feature Sampling. From the analysis in Section 3, we can see that local feature sampling is critical to data efficiency. Fortunately, in detection transformers, the object locations are predicted after each decoder layer. Therefore, we can sample local features under the guidance of the bounding box prediction made by the previous decoder layer without introducing new parameters, as shown in Fig. 2. Although more sophisticated local feature sampling methods can be used, we simply adopt the commonly used RoIAlign [18]. Formally, the sampling operation can be written as:

$$\mathbf{z}_\ell^L = \text{RoIAlign} \left(z_e^L, \mathbf{b}_{\ell-1}\right), \quad \ell = 2 \ldots L_\text{d} \tag{3}$$

where $\mathbf{b}_{\ell-1}$ is the bounding boxes predicted by the previous layer, $\mathbf{z}_\ell^L \in \mathbb{R}^{N \times K^2 \times D}$ is the sampled feature, K is the feature resolution in RoIAlign sampling. Note

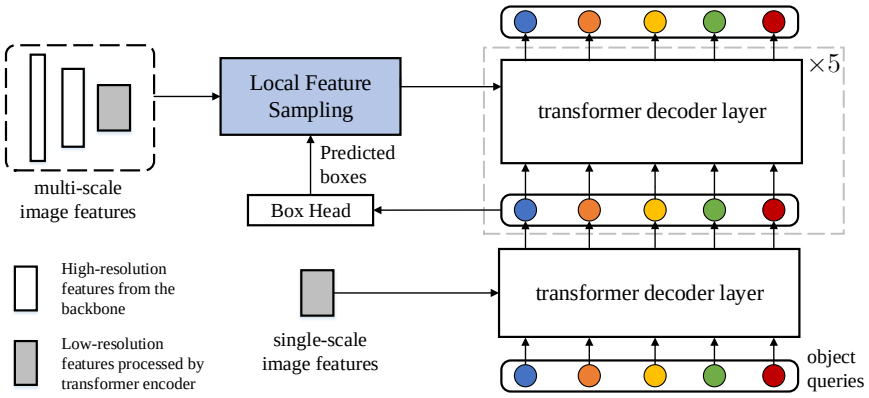

Fig. 2. The proposed data-efficient detection transformer structure. With minimum modifications, we perform sparse sampling feature on key and value feature sent to the cross-attention layers in the decoder, under the guidance of bounding boxes predicted by the previous layer. Note the box head is part of the original detection transformers, which utilize deep supervision on the predictions made by each decoder layer. The backbone, the transformer encoder, and the first decoder layer are kept unchanged.

the reshape and flatten operations are omitted in Equation 3. Similarly, the corresponding positional embedding p_ℓ^L can be obtained.

The cascaded structure in the detection transformer makes it natural to use layer-wise bounding box refinement [1,50] to improve detection performance. Our experiments in Section 3 also validate the effectiveness of the iterative refinement and making predictions with respect to initial spatial references. For this reason, we also introduce bounding box refinement and initial reference points during our implementation, as did in CondDETR [29].

Incorporating Multi-scale Feature. Our sparse feature sampling makes it possible to use multi-scale features in detection transformers with little computation cost. To this end, we also flatten and embed the high-resolution features extracted by the backbone to obtain $\left\{z^l\right\}_{l=1}^{L-1} \in \mathbb{R}^{S^l \times D}$ for local feature sampling. However, these features are not processed by the transformer encoder. Although more sophisticated techniques can be used, these single-scale features sampled by RoIAlign are simply concatenated to form our multi-scale feature. These features are naturally fused by the cross-attention in the decoder.

$$\mathbf{z}_\ell^{\mathrm{ms}} = \left[\mathbf{z}_\ell^1, \mathbf{z}_\ell^2, \ldots, \mathbf{z}_\ell^L\right], \ell = 2 \ldots L_{\mathrm{d}}, \tag{4}$$

where $\mathbf{z}_\ell^{\mathrm{ms}} \in \mathbb{R}^{N \times LK^2 \times D}$ is the multi-scale feature, and $\mathbf{z}_\ell^l = \mathrm{RoIAlign}\left(z^l, \mathbf{b}_{\ell-1}\right), l = 1 \ldots L-1$. The corresponding positional embedding $\mathbf{p}_\ell^{\mathrm{ms}}$ is obtained in a similar way. The decoding process is the same as original detection transformers, as shown in Equation 1, where we have $\mathbf{z}_\ell = \mathbf{z}_\ell^{\mathrm{ms}}$ and $p_\ell = p_\ell^{\mathrm{ms}}$. Please refer to the Appendix for details in implementation.

(a) Bipartite matching in existing detection transformers

(b) Bipartite matching with the proposed label augmentation

Fig. 3. Illustration of the proposed label augmentation method. The predictions and the ground truths are represented by circles and rectangles, respectively. The matching between foreground instances is represented by solid lines, while the matching between background instances is represented by dotted lines. The prediction in blue that was originally matched to a background instance in (a) is now matched to a foreground instance in our method (b), thus obtaining more abundant supervision.

4.3 Label Augmentation for Richer Supervision

Detection transformers perform one-to-one matching for label assignment, which means only a small number of detection candidates are provided with a positive supervision signal in each iteration. As a result, the model has to obtain enough supervision from a larger amount of data or more training epochs.

To alleviate this problem, we propose a label augmentation strategy to provide a richer supervised signal to the detection transformers, by simply repeating positive labels during bipartite matching. As shown in Fig. 3, we repeat the labels of each foreground sample y_i for R_i times, while keeping the total length of the label set N unchanged.

$$y = \left\{ y_1^1, y_1^2, \ldots, y_1^{R_1}, \ldots, y_M^1, y_M^2, \ldots, y_M^{R_M}, \ldots, \varnothing, \ldots, \varnothing \right\}. \tag{5}$$

Subsequently, the label assignment is achieved according to the operation in Equation 2.

Two label repeat strategies are considered during our implementation as follows. (a) Fixed repeat times, where all positive labels are repeated for the same number of times, *i.e.*, $R_i = R, i = 1 \ldots M$. (b) Fixed positive sample ratio, where the positive labels are sampled repeatedly to ensure a proportion of r positive samples in the label set. Specifically, $F = N \times r$ is the expected number of positive samples after repeating labels. We first repeat each positive label for $F//M$ times, and subsequently, randomly sample $F\%M$ positive labels without repetition. By default, we use the fixed repeat times strategy, because it is easier to implement and the resultant label set is deterministic.

5 Experiments

Datasets. To explore detection transformers' data efficiency, most of our experiments are conducted on small-size datasets including Cityscapes [7] and sub-sampled COCO 2017 [24]. Cityscapes contains 2,975 images for training and 500 images for evaluation. For the sub-sampled COCO 2017 dataset, the training

images are randomly sub-sampled by 0.1, 0.05, 0.02, and 0.01, while the evaluation set is kept unchanged. Besides, we also validate the effectiveness of our method on the full-size COCO 2017 dataset with 118K training images.

Implementation details. By default, our feature sampling is implemented as RoIAlign with a feature resolution of 4. Three different feature levels are included for multi-scale feature fusion. A fixed repeat time of 2 is adopted for our label augmentation and non-maximum suppression (NMS) with a threshold of 0.7 is used for duplicate removal. All models are trained for 50 epochs and the learning rate decays after 40 epochs, unless specified. ResNet-50 [19] pre-trained on ImageNet-1K [9] is used as backbone. To guarantee enough number of training iterations, all experiments on Cityscapes and sub-sampled COCO 2017 datasets are trained with a batch size of 8. And the results are averaged over five repeated runs with different random seeds. Our data-efficient detection transformers only make slight modifications to existing methods. Unless specified, we follow the original implementation details of corresponding baseline methods [3,29]. Run time is evaluated on NVIDIA A100 GPU.

5.1 Main Results

Table 2. Comparison of detection transformers on Cityscapes. DE denotes data-efficient and LA denotes label augmentation. † indicates the query number is increased from 100 to 300.

Method	Epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params	FLOPs	FPS
DETR [3]	300	11.5	26.7	8.6	2.5	9.5	25.1	41M	86G	44
UP-DETR [8]	300	23.8	45.7	20.8	4.0	20.3	46.6	41M	86G	44
PnP-DETR-α=0.33 [39]	300	11.2	11.5	8.7	2.3	21.2	25.6	41M	79G	43
PnP-DETR-α=0.80 [39]	300	11.4	26.6	8.1	2.5	9.3	24.7	41M	83G	43
CondDETR [29]	50	12.1	28.0	9.1	2.2	9.8	27.0	43M	90G	39
SMCA (single scale) [14]	50	14.7	32.9	11.6	2.9	12.9	30.9	42M	86G	39
DeformDETR [50]	50	27.3	49.2	26.3	8.7	28.2	45.7	40M	174G	28
DE-DETR	50	21.7	41.7	19.2	4.9	20.0	39.9	42M	88G	34
DELA-DETR†	50	24.5	46.2	22.5	6.1	23.3	43.9	42M	91G	29
DE-CondDETR	50	26.8	47.8	25.4	6.8	25.6	46.6	44M	107G	29
DELA-CondDETR	50	29.5	52.8	27.6	7.5	28.2	50.1	44M	107G	29

Results on Cityscapes. In this section, we compare our method with existing detection transformers. As shown in Table 2, most of them suffer from the data-efficiency issue. Nevertheless, with minor changes to the CondDETR model, our DE-CondDETR is able to achieve comparable data efficiency to DeformDETR. Further, with the richer supervision provided by label augmentation, our DELA-CondDETR surpasses DeformDETR by 2.2 AP. Besides, our method can be combined with other detection transformers to significantly improve their data efficiency, for example, our DE-DETR and DELA-DETR trained for 50 epochs perform significantly better than DETR trained for 500 epochs.

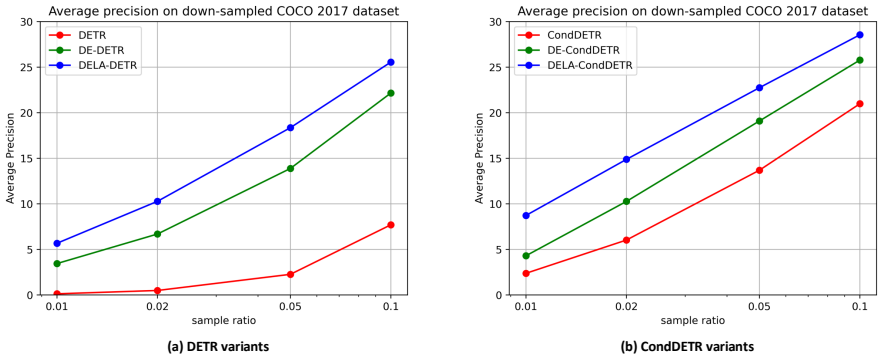

Fig. 4. Performance comparison of different methods on sub-sampled COCO 2017 dataset. Note the sample ratio is shown on a logarithmic scale. As can be seen, both local feature sampling and label augmentation consistently improve the model performance under varying data sampling ratios.

Results on sub-sampled COCO 2017. Sub-sampled COCO 2017 datasets contain 11,828 (10%), 5,914 (5%), 2,365 (2%), and 1,182 (1%) training images, respectively. As shown in Fig 4, our method consistently outperforms the baseline methods by a large margin. In particular, DELA-DETR trained with only ∼1K images significantly outperforms the DETR baseline with five times the training data. Similarly, DELA-CondDETR consistently outperforms the CondDETR baseline trained with twice the data volume.

5.2 Ablations

In this section, we perform ablated experiments to better understand each component of our method. All the ablation studies are implemented on the DELA-CondDETR and the Cityscapes dataset, while more ablation studies based on DELA-DETR can be found in our Appendix.

Table 3. Ablations on each component in DELA-CondDETR. "SF", "MS", and "LA" represent sparse feature sampling, multi-scale feature fusion, and label augmentation.

Method	SF	MS	LA	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params	FLOPs	FPS
CondDETR [29]				12.1	28.0	9.1	2.2	9.8	27.0	43M	90G	39
			✓	14.7	31.6	12.1	2.9	12.5	32.1	43M	90G	38
	✓			20.4	40.7	17.7	2.9	16.9	42.0	44M	95G	32
DE-CondDETR	✓	✓		26.8	47.8	25.4	6.8	25.6	46.6	44M	107G	29
DELA-CondDETR	✓	✓	✓	29.5	52.8	27.6	7.5	28.2	50.1	44M	107G	29

Effectiveness of each module. We first ablate the role of each module in our method, as shown in Table 3. The use of local feature sampling and multi-

scale feature fusion significantly improves the performance of the model by 8.3 and 6.4 AP, respectively. In addition, label augmentation further improves the performance by 2.7 AP. Besides, using it alone also brings a gain of 2.6 AP.

Table 4. Ablations on multi-scale feature levels and feature resolutions for RoIAlign. 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	14.8	35.1	11.0	2.4	11.7	31.1	44M	90G	32
1	4	20.4	40.7	17.7	2.9	16.9	42.0	44M	95G	32
1	7	20.7	40.9	18.5	2.9	16.8	42.7	44M	104G	31
3	4	26.8	47.8	25.4	6.8	25.6	46.6	44M	107G	29
4	4	26.3	47.1	25.1	6.5	24.8	46.5	49M	112G	28

Feature resolution for RoIAlign. In general, a larger sample resolution in RoIAlign provides richer information and thus improves detection performance. However, sampling larger feature resolution is also more time-consuming and increases the computational cost of the decoding process. As shown in Table 4, the model performance is significantly improved by 5.6 AP when the resolution is increased from 1 to 4. However, when the resolution is further increased to 7, the improvement is minor and the FLOPs and latency are increased. For this reason, we set the feature resolution for RoIAlign as 4 by default.

Number of multi-scale features. To incorporate multi-scale features, we also sample the 8× and 16× down-sampled features from the backbone to construct multi-scale features of 3 different levels. As can be seen from Table 4, it significantly improves the model performance by 6.4 AP. However, when we further add the 64× down-sampled features for multi-scale fusion, the performance drops by 0.5 AP. By default, we use 3 feature levels for multi-scale feature fusion.

Strategies for label augmentation. In this section, we ablate the proposed two label augmentation strategies, namely fixed repeat time and fixed positive sample ratio. As shown in Table 5, using different fixed repeated times consistently improves the performance of DE-DETR baseline, but the performance gain tends to decrease as the number of repetitions increases. Moreover, as shown

Table 5. Ablations on label augmentation using fixed repeat time. Params, FLOPs, and FPS are omitted since they are consistent for all settings.

Time	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
–	26.8	47.8	25.4	6.8	25.6	46.6
2	29.5	52.8	27.6	7.5	28.2	50.1
3	29.4	52.6	28.0	7.6	28.1	50.3
4	29.0	52.0	27.7	7.8	27.9	49.5
5	28.7	51.3	27.4	7.8	27.7	49.3

Table 6. Ablations on label augmentation using fixed positive sample ratio.

Ratio	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
–	26.8	47.8	25.4	6.8	25.6	46.6
0.1	27.7	49.7	26.1	7.4	26.5	47.2
0.2	28.2	50.2	26.9	7.4	26.8	48.5
0.25	28.3	50.5	27.2	7.5	27.1	48.3
0.3	27.9	50.3	26.5	7.3	27.1	47.4
0.4	27.6	49.7	26.0	7.0	27.0	46.8

in Table 6, although using different ratios can bring improvement on AP, the best performance is achieved when the positive to negative samples ratio is 1:3, which, interestingly, is also the most commonly used positive to negative sampling ratio in the RCNN series detectors, *e.g.* Faster RCNN.

Table 7. Performance of our data-efficient detection transformers on COCO 2017. All models are trained for 50 epochs.

Method	Epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params	FLOPs	FPS
DETR [3]	50	33.6	54.6	34.2	13.2	35.7	53.5	41M	86G	43
DE-DETR	50	40.2	60.4	43.2	23.3	42.1	56.4	43M	88G	33
DELA-DETR†	50	41.9	62.6	44.8	24.9	44.9	56.8	43M	91G	29
CondDETR [29]	50	40.2	61.1	42.6	19.9	43.6	58.7	43M	90G	39
DE-CondDETR	50	41.7	62.4	44.9	24.4	44.5	56.3	44M	107G	28
DELA-CondDETR	50	43.0	64.0	46.4	26.0	45.5	57.7	44M	107G	28

5.3 Generalization to Sample-Rich Dataset

Although the above experiments show that our method can improve model performance when only limited training data is available, there is no guarantee that our method remains effective when the training data is sufficient. To this end, we evaluate our method on COCO 2017 with a sufficient amount of data. As can be seen from Table 7, our method does not degrade the model performance on COCO 2017. Conversely, it delivers a promising improvement. Specifically, DELA-DETR and DELA-CondDETR improve their corresponding baseline by 8.3 and 2.8 AP, respectively.

6 Conclusion

In this paper, we identify the data-efficiency issue of detection transformers. Through step-by-step model transformation from Sparse RCNN to DETR, we find that sparse feature sampling from local areas holds the key to data efficiency. Based on these, we improve existing detection transformers by simply sampling multi-scale features under the guidance of predicted bounding boxes, with minimum modifications to the original models. In addition, we propose a simple yet effective label augmentation strategy to provide richer supervision and thus further alleviate the data-efficiency issue. Extensive experiments validate the effectiveness of our method. As transformers become increasingly popular for visual tasks, we hope our work will inspire the community to explore the data efficiency of transformers for different tasks.

Acknowledgement. This work is supported by National Key R&D Program of China under Grant 2020AAA0105701, National Natural Science Foundation of China (NSFC) under Grants 61872327, Major Special Science and Technology Project of Anhui (No. 012223665049), and the ARC project FL-170100117.

References

1. Cai, Z., Vasconcelos, N.: Cascade r-cnn: Delving into high quality object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6154–6162 (2018)
2. Cao, Y.H., Yu, H., Wu, J.: Training vision transformers with only 2040 images. arXiv preprint arXiv:2201.10728 (2022)
3. 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)
4. Chen, Z., Zhang, J., Tao, D.: Recurrent glimpse-based decoder for detection with transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5260–5269 (2022)
5. Chen, Z., Xie, L., Niu, J., Liu, X., Wei, L., Tian, Q.: Visformer: The vision-friendly transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 589–598 (2021)
6. Chu, X., Tian, Z., Wang, Y., Zhang, B., Ren, H., Wei, X., Xia, H., Shen, C.: Twins: Revisiting the design of spatial attention in vision transformers. Advances in Neural Information Processing Systems **34** (2021)
7. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3213–3223 (2016)
8. Dai, Z., Cai, B., Lin, Y., Chen, J.: Up-detr: Unsupervised pre-training for object detection with transformers. In: Proceedings of the IEEE conference on computer vision and pattern recognition (2020)
9. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. IEEE (2009)
10. 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)
11. Fang, J., Xie, L., Wang, X., Zhang, X., Liu, W., Tian, Q.: Msg-transformer: Exchanging local spatial information by manipulating messenger tokens. arXiv preprint arXiv:2105.15168 (2021)
12. Fang, Y., Liao, B., Wang, X., Fang, J., Qi, J., Wu, R., Niu, J., Liu, W.: You only look at one sequence: Rethinking transformer in vision through object detection. arXiv preprint arXiv:2106.00666 (2021)
13. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE transactions on pattern analysis and machine intelligence **32**(9), 1627–1645 (2009)
14. 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)
15. Ge, Z., Liu, S., Li, Z., Yoshie, O., Sun, J.: Ota: Optimal transport assignment for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 303–312 (2021)
16. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 580–587 (2014)

17. Han, K., Xiao, A., Wu, E., Guo, J., Xu, C., Wang, Y.: Transformer in transformer. Advances in Neural Information Processing Systems **34** (2021)
18. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
20. Jia, X., De Brabandere, B., Tuytelaars, T., Gool, L.V.: Dynamic filter networks. Advances in neural information processing systems **29** (2016)
21. Lee, C.Y., Xie, S., Gallagher, P., Zhang, Z., Tu, Z.: Deeply-supervised nets. In: Artificial intelligence and statistics. pp. 562–570. PMLR (2015)
22. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2117–2125 (2017)
23. 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)
24. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision (2014)
25. Liu, F., Wei, H., Zhao, W., Li, G., Peng, J., Li, Z.: Wb-detr: Transformer-based detector without backbone. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2979–2987 (2021)
26. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: European conference on computer vision. pp. 21–37. Springer (2016)
27. 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)
28. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. arXiv preprint arXiv:2103.14030 (2021)
29. 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)
30. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788 (2016)
31. 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)
32. Shen, Y., Wang, X., Tan, Z., Xu, G., Xie, P., Huang, F., Lu, W., Zhuang, Y.: Parallel instance query network for named entity recognition. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics (2022), https://arxiv.org/abs/2203.10545
33. 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)
34. Tian, Z., Shen, C., Chen, H.: Conditional convolutions for instance segmentation. In: European Conference on Computer Vision. pp. 282–298. Springer (2020)

35. Tian, Z., Shen, C., Chen, H., He, T.: Fcos: Fully convolutional one-stage object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9627–9636 (2019)
36. 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)
37. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Conference on Neural Information Processing Systems (2017)
38. Wang, J., Chen, K., Yang, S., Loy, C.C., Lin, D.: Region proposal by guided anchoring. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2965–2974 (2019)
39. 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)
40. Wang, W., Cao, Y., Zhang, J., Tao, D.: FP-DETR: Detection transformer advanced by fully pre-training. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=yjMQuLLcGWK
41. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 568–578 (2021)
42. Xu, Y., Zhang, Q., Zhang, J., Tao, D.: Vitae: Vision transformer advanced by exploring intrinsic inductive bias. Advances in Neural Information Processing Systems **34** (2021)
43. Yang, T., Zhang, X., Li, Z., Zhang, W., Sun, J.: Metaanchor: Learning to detect objects with customized anchors. Advances in neural information processing systems **31** (2018)
44. Yuan, H., Li, X., Yang, Y., Cheng, G., Zhang, J., Tong, Y., Zhang, L., Tao, D.: Polyphonicformer: Unified query learning for depth-aware video panoptic segmentation. In: European Conference on Computer Vision (2022)
45. Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z.H., Tay, F.E., Feng, J., Yan, S.: Tokens-to-token vit: Training vision transformers from scratch on imagenet. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 558–567 (2021)
46. Zhang, Q., Xu, Y., Zhang, J., Tao, D.: Vitaev2: Vision transformer advanced by exploring inductive bias for image recognition and beyond. arXiv preprint arXiv:2202.10108 (2022)
47. Zhang, S., Chi, C., Yao, Y., Lei, Z., Li, S.Z.: Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9759–9768 (2020)
48. Zhang, X., Wan, F., Liu, C., Ji, R., Ye, Q.: Freeanchor: Learning to match anchors for visual object detection. Advances in neural information processing systems **32** (2019)
49. Zhu, B., Wang, J., Jiang, Z., Zong, F., Liu, S., Li, Z., Sun, J.: Autoassign: Differentiable label assignment for dense object detection. arXiv preprint arXiv:2007.03496 (2020)
50. 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)