WB-DETR: Transformer-Based Detector without Backbone

Fanfan Liu[1,2,3,4]*, Haoran Wei[1,2,3,4]*, Wenzhe Zhao[1,2,3,4]†, Guozhen Li[5],
Jingquan Peng[1,2,3,4], Zihao Li[1,2,3,4]
[1]Aerospace Information Research Institute, Chinese Academy of Sciences.
[2]NIST, Aerospace Information Research Institute, Chinese Academy of Sciences.
[3]University of Chinese Academy of Sciences, Beijing, China.
[4]School of Electronic, Electrical and Communication Engineering, UCAS.
[5]Dalian University of Technology, Dalian, China.
(liufanfan19, weihaoran18, pengjingquan19, lizihao191)@mails.ucas.ac.cn,
zwz@mail.ie.ac.cn, lgzh@mail.dlut.edu.cn

Abstract

Transformer-based detector is a new paradigm in object detection, which aims to achieve pretty-well performance while eliminates the priori knowledge driven components, e.g., anchors, proposals and the NMS. DETR, the state-of-the-art model among them, is composed of three sub-modules, i.e., a CNN-based backbone and paired transformer encoder-decoder. The CNN is applied to extract local features and the transformer is used to capture global contexts. This pipeline, however, is not concise enough. In this paper, we propose ***WB-DETR*** *(****DETR****-based detector* ***W****ithout* ***B****ackbone) to prove that the reliance on CNN features extraction for a transformer-based detector is not necessary. Unlike the original DETR, WB-DETR is composed of only an encoder and a decoder without CNN backbone. For an input image, WB-DETR serializes it directly to encode the local features into each individual token. To make up the deficiency of transformer in modeling local information, we design an LIE-T2T (local information enhancement tokens to token) module to enhance the internal information of tokens after unfolding. Experimental results demonstrate that WB-DETR, the first pure-transformer detector without CNN to our knowledge, yields on par accuracy and faster inference speed with only half number of parameters compared with DETR baseline.*

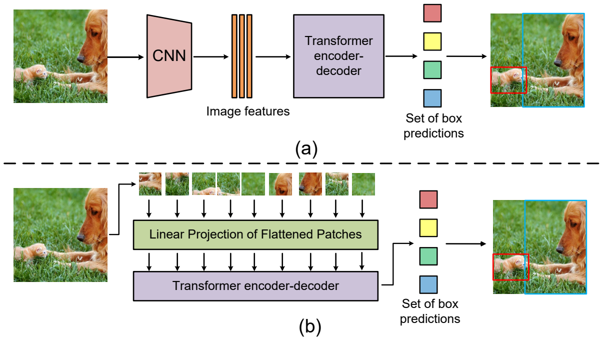
CNN
Image features
Transformer encoder-decoder
Set of box predictions
(a)
Linear Projection of Flattened Patches
Transformer encoder-decoder
Set of box predictions
(b)

Figure 1. DETR vs. WB-DETR. (a) DETR first uses a CNN network to extract features, and then utilizes a transformer structure for object detection. (b) WB-DETR serializes the image and uses transformer to detect object directly.

1. Introduction

CNN-based approaches [18] have dominated object detection tasks [20, 32] for years. In these methods, a common component is the backbone network [12, 13, 14, 35], acting as extracting image features by a series of convolution and pooling layers. Modern CNN-based detectors [9, 11, 27, 21, 36, 29, 23, 25, 26, 22] regard the detector design as a modules combination process, which always composed of a backbone, a neck [21] and multiple detection heads [3]. Among which, the backbone has become a de facto standard to improve the performance and the design of various backbones is also a focus of research in the field of object detection. As we all know, the equipment of a backbone is essential for existing CNN-based detectors.

To get out of the paradigm of CNN-based design, Carion et al. propose a novel detector named DETR [4]. Unlike previous CNN-based works, DETR is a transformer-based detector [4, 40, 5, 33], which eliminates many handcrafted operations [4], e.g., anchor generation, rule-based object assignment, non-maximum suppression (NMS) postprocessing, and so on. As shown in Figure 1 (a), DETR

*Equal contribution.

†Corresponding author is Wenzhe Zhao.

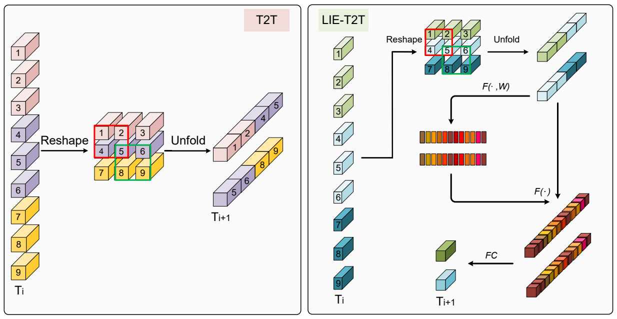

Figure 2. T2T vs. LIE-T2T. T2T aggregates the information of adjacent tokens through reshape and unfold operations. Based on T2T, LIE-T2T can realize local spatial attention of reshaped T_i by calculating channel attention of unfolded T_{i+1}. $F(\cdot, W)$ means an attention calculation, $F(\cdot)$ represents element-wise multiplication and FC indicates the FC layer.

applies a simple architecture that combined with a CNN backbone and paired transformer [31] encoder-decoder to output set of box predictions, which simplifies the pipeline of object detection in an extent. However, DETR is also influenced by the modular splicing design and still relies on CNN to extract features, which makes the model not unify and neat enough.

Vision Transformer (ViT) [6] is the first pure-transformer model that can be directly applied for image classification. It splits the input image into 16 × 16 patches with fixed length. Then, an encoder sub-module is run to conduct sequence modeling of patches to obtain classification results. Unfortunately, ViT achieves inferior performance compared with CNN [12, 13, 14, 35], since the simple tokenization of input images fails to model the important local structure (e.g., edges, lines) among neighboring pixels. T2T-ViT (Tokens-to Token Vision Transformer) [37] solves the above problem by recursively aggregating neighboring tokens into one token. In this way, not only the local structure presented by surrounding tokens that can be modulated, the tokens length also can be reduced. The performance of T2T-VIT exceeds that of the classifier designed by CNN, which proves that transformer is also capable of extracting shallow features. And thus, a nature problem is: *is the CNN-backbone in DETR redundant?*

In this paper, we show the above answer is affirmative. Inspired by [37], we try to get rid of the backbone of DETR and propose what we believe the most concise detector (WB-DETR) so far. As depicted in Figure 1 (b), WB-DETR does not use the backbone of CNN to extract features. Instead, it directly serializing the image, encoding local features in each independent token. As we all know, the self-attention of transformer has strong global information modeling ability, which can commendably modulate the contexts between different tokens. However, the local information in each token and the information between adjacent tokens in space are not well modeled. In other words, transformer lacks the ability of local information modeling. Although the T2T [37] module can aggregate the contexts of adjacent tokens, it is unable to model the internal information of the aggregated independent token separately, as illustrated in Figure 2 (a). Accordingly, along with WB-DETR, we present LIE-T2T (Local Information Enhancement-T2T) module. As shown in Figure 2 (b), LIE-T2T not only reorganizes and unfolds the adjacent tokens, but also calculates the attention on the channel-dimension of each token after unfolding. Because the tokens are obtained from feature map through unfold operation, modeling the relationship between channels of the tokens is equivalent to modeling the spatial relationship between the pixels in feature map. That is why channel attention in LIE-T2T can enhance local information.

In a word, we propose WB-DETR (DETR-Based Detector without Backbone), which is only composed of an encoder and a decoder without the backbone. Instead of utilizing a CNN to extract features, WB-DETR serializes the image directly, encoding the local features of input into each individual token. Besides, to allow WB-DETR better make up the deficiency of transformer in modeling local information, we design LIE-T2T (Local Information Enhancement Tokens-to Token) module to modulate the internal (local) information of each token after unfolding. Compared with the DETR baseline, WB-DETR without backbone is more unify and neat. We encourage researchers to rethink the modules combination (backbone-neck-head) design paradigm for object detection.

2. Related Works

2.1. Object Detection

Object detection is a basic task in computer vision, which operates as localizing and classifying objects in an image. With the assistance of deep learning, object detection yields great progress in the current era, promoting broad vision tasks directly or indirectly, such as object tracking [2, 15], instance segmentation [11], pose estimation [8, 24], and so on. Modern object detectors hammer at maintaining the high precision in the process of pursuing pipeline simplicity. Two-stage detectors [28, 3]predict boxes, w.r.t., proposals, whereas single-stage methods make predictions, w.r.t., anchors [22] or grids of possible objects centers [39, 29]. In the last few years, anchor box is used to match a ground truth box and acts as a guidance for detectors to regress the object bounding box. Faster R-CNN [28] popularizes the anchor mechanism in its Region Proposal Network (RPN) which used to generate proposals from a set of candidate boxes. Later, the anchor boxes are widely used in the two-stage and anchor-based detectors. Afterwards, to further explore the efficiency of mod-

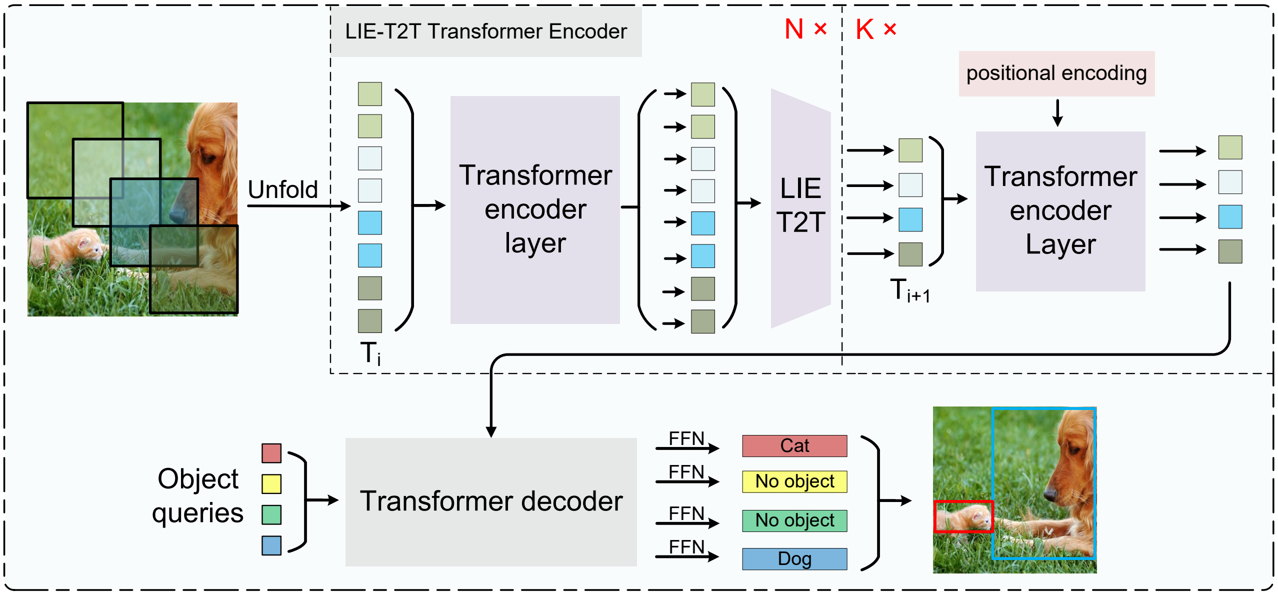

Figure 3. The architecture of the proposed WB-DETR. Firstly, an input image is soft split as patches, and unfolded as a sequence of tokens T_0, Then, T_0 is fed into the LIE-T2T Transformer Encoder, composed of N layers LIE encoders and K layers encoders without LIE, to get T_i. Finally, WB-DETR utilizes each output embedding of the decoder to a shared Feed Forward Network (FFN) to predict either an "object" (with class and bounding box) or a "no object".

els, some anchor-based one-stage detectors also appeared. They remove the RPN and directly regress and classify the anchor boxes. YOLOv2 [26] uses anchor boxes to predict bounding boxes, which achieves much better performance than YOLOv1[25]. Recently, considering the drawbacks of the anchor mechanism, researchers have proposed many anchor-free methods. FCOS [29] treats pixels as positive samples and directly regresses four-vectors (the distances from each pixel to the borders of the corresponding box). In addition, Keypoints-based detectors usually predict keypoints via outputting heatmaps [30]. For example, CornerNet [19] detects objects by predicting and grouping pairs of corner points. Based on CornerNet, Duan *et al.* designed CenterNet [7] that detects each object as a triplet. Recent works [38] demonstrate that the final performances of the above models heavily depend on the exact way of what initial guesses are set, e.g., the settings of anchors and the matching rules of positive and negative samples. When these detectors match positive and negative samples, they often match multiple predictions with one target, which lead to one object along with multiple bounding boxes predictions. They require prior knowledge, such as NMS, to filter out excess boxes. That is why the object detector cannot be designed as complete end-to-end.

2.2. Visual Transformers

The concept of transformers is first proposed in [31] for the sequence-to-sequence machine translation task, and since then transformers have become the de facto method in most NLP (Natural Language Processing) tasks [1]. As the core mechanism of transformers, self-attention is particularly suitable for modeling long-range dependencies. Recently, transformers start to show prospects in computer vision tasks. DETR [4] builds an object detection systems based on transformers, which largely simplifies the traditional detection pipeline, and achieves on par performances compared with highly optimized CNN-based detectors. ViT [6] introduces the transformer to image recognition and models an image as a sequence of patches, which attains excellent results compared with state-of-the-art CNN networks. The above two works show the effectiveness of transformers in image understanding tasks. Our work is inspired by DETR and ViT. To our knowledge, there is no object detector uses pure-transformer without any CNN modules. *Whether the object detection task can be completed using only transformer?* In this paper, we introduce the WB-DETR and provide an affirmative answer to it.

3. Method

In this section, we first introduce the overall pipeline of the proposed WB-DETR. Next, we will delve into each module of the proposed WB-DETR and show how LIE-T2T helps better model local information. Finally, we introduce the design of loss functions.

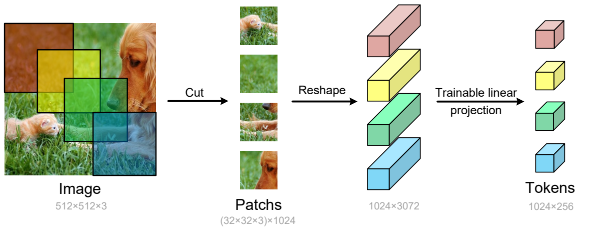

Figure 4. The process of Image to Tokens. Take an input image with 512×512×3 as an example. Firstly, the image is cut to 1024 patches with the size of 32×32 × 3. Then, each patch is reshaped to one-dimensional. Finally, a trainable linear projection is performed to yield required tokens.

3.1. Image to tokens

We follow the ViT to handle 2D images. Firstly, We cut the image to a size of (p, p) with a step size of (s, s). In this way, the input image $x \in R^{h\times w\times c}$ is reshaped into a sequence of flattened 2D patches $\mathrm{x}_p \in R^{l\times c_p}$, where h and w are the height and width of the original image, c is the number of channels, and l represents the length of patch. Among them, $l = \frac{h\times w}{s^2}$, $c_p = p^2 \times c$. l also serves as the effective input sequence length for the transformer encoder. Our LIE-T2T encoder employs constant latent vector size d through all of its layers. And thus, we flatten and map the patches to d dimensions with a trainable linear projection. More specifically, this linear projection has an input and output dimensions of c_p and d, respectively. We name the output of this projection as the tokens T_0. The procedure of converting image to tokens is shown in Figure 4.

3.2. LIE-T2T encoder

After the process of image to tokens, we add positional encodings [4] to target tokens to make them carry location information. The positional encoding is a standard learnable 1D version [6, 3]. Then, the resulting sequence of embedding vectors serves as input to the encoder, as shown in Figure 3. Each encoder layer keeps a standard architecture which consists of a multi-head self-attention module and a feed forward network (FFN). An LIE-T2T module is equipped behind each encoder layer to constitute the LIE-T2T encoder. The LIE-T2T module can progressively reduce the length of tokens and transform the spatial structure of the image.

Since we do not use any CNN-based backbone to extract image features, instead of directly serializing the image, the local information of the image is encoded in each independent token. Although the self-attention sub-module in transformer has strong global information modeling capabilities, which can model information between different tokens, the local information within each token and the information between adjacent tokens in space are not well modeled. To this end, when designing the LIE-T2T module, we need to not only reorganize and stretch the adjacent tokens, but also enhance the internal information (i.e., local information) of the tokens after flattening.

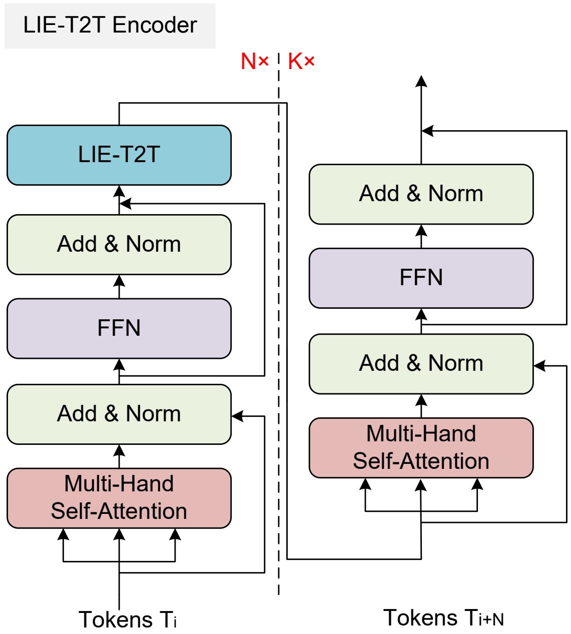

Figure 5. Detailed structure diagram of the LIE-T2T encoder. Each encoder layer keeps a standard architecture which consists of a multi-head self-attention module and a feed forward network (FFN). An LIE-T2T module is equipped behind each encoder layer to constitute the LIE-T2T encoder.

Concretely, LIE-T2T module calculates attention on the channel-dimension of each token. The attention is calculated separately for each token. More detailed iterative process of LIE-T2T module is shown in Figure 5, which can also be formulated as follows:

$$T = Unfold\left(Reshape\left(T_i\right)\right) \tag{1}$$

$$S = Sigmoid\left(W_2 \cdot ReLU\left(W_1 \cdot T\right)\right) \tag{2}$$

$$T_{i+1} = W_3 \cdot (T \cdot S) \tag{3}$$

where $Reshape$ means the operation: reorganize $(l_1 \times c_1)$ tokens into $(h \times w \times c)$ feature map. $Unfold$ represents stretching $(h \times w \times c)$ feature map to $(l_2 \times c_2)$ tokens. W_1, W_2, and W_3 indicate parameters of corresponding fully connected layer. We use the $ReLU$ activation to find its nonlinear mapping and employ the $Sigmoid$ function to generate the final attention. The input of LIE-T2T encoder is with the dimension of $((h/s \times w/s) \times 256)$.

3.3. Decoder

The decoder of WB-DETR follows the standard architecture of the transformer [4], acting as transforming N embeddings of size d using multi-headed self-attention and encoder-decoder mechanisms. Like DETR, our WB-DETR decodes the N objects in parallel at each decoder layer. Since the decoder is permutation-invariant, the N input embeddings must be distinguishable to produce different results. These input embeddings are learnable embeddings that we refer to object queries. As the encoder, we add positional encoding to the input of each attention layer in decoder. Finally, the N object queries are transformed into an output embedding by the decoder and then independently decoded into box coordinates and class labels by a feed forward network (FFN), yielding N final predictions.

3.4. Feed Forward Network

The feed forward network is computed by a 3-layer perceptron with a $ReLU$ activation function and a linear projection layer. The final outputs of FFN are the normalized center coordinates, height and width of the boxes w.r.t. the input image, and the linear layer predicts the class label via a $Softmax$ function. Since we predict a fixed-size set of N bounding boxes, where N is usually much larger than the actual number of objects of interest in an image, an additional special class label ”no object” [4] is used to represent that no object is detected within a slot. This ”no object” class plays a similar role to the “background” class in the traditional object detection approaches [11, 23].

3.5. Loss Functions

The loss functions of WB-DETR are the same as DETR, which are driven by Hungarian algorithm. In other words, all supervisions are applied after the matching between predictions and ground-truths.

Matching. Our loss functions produce an optimal bipartite matching between predicted and ground-truth objects. We use the Hungarian algorithm to find an optimal match and the matching cost is composed of predicted class and bounding box following [4]. After matching, we can get a new order of ground-truth objects, and then multi-classification loss and bounding box loss are calculated based on the new matching ground-truth.

Multi-classification loss. WB-DETR adopts cross-entropy loss with balanced-weights as multi-classification loss function. The specific formula can be expressed as:

$$\mathcal{L}_c(y,\hat{y}) = -\frac{1}{N}\sum_{i=1}^{i=N}\begin{cases}\log(\hat{y}^i) & \text{if } y_{class} \neq \text{no object}\\ \alpha\cdot\log(\hat{y}^i) & \text{otherwise,}\end{cases} \tag{4}$$

where α is the loss weight, balancing the object and “no object” samples, which we set to 0.1.

Bounding box loss. The regression loss of bounding box consists of two parts: L1 loss and IoU loss as follows.

$$L_{box}(\hat{b},b) = \frac{1}{N}\sum_{i=1}^{i=N}[\gamma\cdot L_1(\hat{b}_i,b_i)+ \\ \eta\cdot L_{iou}(\hat{b}_i,b_i)] \tag{5}$$

where γ and η are the balanced-weights of L_1 and L_{iou}. $\hat{b}$ and b represent the regressed and ground-truth bounding box, respectively.

4. Experiments

4.1. Dataset

We evaluate the proposed WB-DETR on the very challenge MS COCO benchmark dataset [20, 17]. We train our model on the train2017 split with about $115K$ annotated images and validate our method on the val2017 split with $5K$ images. COCO uses average precision (AP) at different IoUs as the main evaluation metrics.

4.2. Implementation details

The main settings and training strategy of our WB-DETR are mainly followed DETR [4] for better comparisons. All transformer weights are initialized with Xavier Init [10], and our model has no pre-train process on any external dataset. By default, models are trained for 500 epochs with a learning rate drop $10\times$ at the 400 epoch. We optimize WB-DETR via an Adam optimizer [16] with a base learning rate of $1e{-}4$ and a weight decay of 0.001. We use a batch size of 32 and train the network on 16 V100 GPUs with 4 images per-GPU. We use some standard data augmentations, such as random resizing, color jittering, random flipping and so on to overcome the overfitting. The transformer is trained with a default dropout of 0.1. We fix the number of decoding layers at 6 and report performance with the different layer number N and K of encoder: When N and K is n and k, the corresponding model is named as WB-DETR$_k^n$.

4.3. Comparisons with Faster R-CNN and DETR

We validate the effectiveness of WB-DETR by comparing it with the most classic detectors (Faster R-CNN and DETR). As shown in Table 1, our WB-DETR (2-12) model yields on par AP with DETR while the number of parameters of our model is only about half of the DETR. Besides, the speed is 8 FPS faster than that of DETR. The WB-DETR (2-8) achieves the similar AP (40.2) and about twice of the inference speed also only with half of parameters compared with Faster R-CNN (with FPN) model. The above results prove that without a CNN-based backbone, a pure-transformer is capable of all the steps of object detection task, and even better than the CNN-based ones.

Table 1. Comparison with Faster R-CNN and DETR on the COCO validation set. The parameters of our model are greatly reduced, and the inference speed is a lot faster than the classic Faster R-CNN and DETR.

Method	Params	FLOPs	FPS	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster R-CNN w/FPN	42M	180G	26	40.2	61.0	43.8	24.2	43.5	52.0
Faster R-CNN-DC5	166M	320G	16	39.0	60.5	42.3	21.4	43.5	52.5
Faster R-CNN-R101 w/FPN	60M	246G	20	42.0	62.5	45.9	25.2	45.6	54.6
DETR	41M	86G	28	42.0	62.4	44.2	20.5	45.8	61.1
DETR-DC5	41M	187G	12	43.3	63.1	45.9	22.5	47.3	61.1
DETR-R101	60M	152G	20	43.5	63.8	46.4	21.9	48.0	61.8
DETR-DC5-R101	60M	253G	10	44.9	64.7	47.7	23.7	49.5	62.3
WB-DETR (2-4)	14M	62G	51	39.6	58.4	43.8	18.2	42.7	54.9
WB-DETR (2-8)	19M	80G	42	40.2	60.1	43.9	19.3	44.1	58.8
WB-DETR (2-12)	24M	98G	36	41.8	63.2	44.8	19.4	45.1	62.4

Table 2. We evaluated the effectiveness of our proposed LIE-T2T module by changing the number of LIE-T2T encoder layers.

Layers	Params	FLOPs	FPS	AP	AP_S	AP_M	AP_L
0	13M	60G	49	30.7	10.5	35.2	56.5
1	16M	72G	45	38.5	17.1	40.8	57.6
2	19M	80G	42	40.2	19.3	44.1	58.8
3	21M	94G	40	40.3	19.2	44.3	58.7

Table 3. Comparison with LIE-T2T and T2T. The values of AP_{50} yielded by LIE-T2T and T2T are very close, and the AP_{75} achieved by LIE-T2T is much higher than T2T.

LIE-T2T	T2T	Params	FLOPs	FPS	AP	AP_{50}	AP_{75}
	✓	19M	79G	42	38.1	62.8	41.4
✓		19M	80G	42	40.2	63.2	44.8

4.4. Ablation Studies

In the transformer decoder, self-Attention is the key component which models relations between feature representations of different predictions. In the above experiments, the number of decoder layers is fixed at 6 as default. In this part, with the fixed decoder, we explore how other components of our architecture and loss functions influence the final performance. All ablation studies only use WB-DETR (2-8) for limited computing resources.

Number of LIE-T2T encoder layers. We evaluate the effectiveness of the proposed LIE-T2T module further by changing the number of LIE-T2T encoder layers. More specifically, when we reduce the number of LIE-T2T encoder layers, we also increase the step size of each layer to ensure that the subsequent dimensions are consistent. As illustrated in Table 2, we can see that if we do not add our LIE-T2T module to original encoder, the AP would drop significantly (from 40.3 to 30.7), especially for small targets, with a drop of nearly $\frac{1}{2}$ AP (from 19.2 to 10.5). This result can adequately demonstrate that our LIE-T2T encoder layers can make the transformer process local information better. As the number of LIE-T2T encoder layers continues to increase (from 1 to 2), the delta AP is getting smaller. 2 LIE-T2T encoder layers achieves on par AP (40.2 vs.40.3) with 3. That is why we select 2 as the number of LIE-T2T encoder layers. Of course, in other view, this experiment also shows that a pure-transformer without any modification can still perform rough object detection.

LIE-T2T vs. T2T. We compare our proposed LIE-T2T with the original T2T module. Although T2T module can aggregate the information of adjacent tokens, it can not model the internal information of the aggregated independent tokens separately. Accordingly, we designed the LIE-T2T module based on T2T. As mentioned above, LIE-T2T not only reorganizes and stretches the adjacent tokens, but also calculates the attention in the channel-dimension of each token after stretching. Because the tokens are obtained by feature map through unfolding, modeling the relationship between the channels of tokens is equivalent to modeling the spatial relationship between the pixels. We add channel-dimension attention on tokens, which is equivalent to add local spatial attention. It can be seen from Table 3 that the values of AP_{50} yielded by LIE-T2T and T2T are very close, yet the AP_{75} achieved by LIE-T2T is much higher than T2T. The experiment result shows that the regressed bounding boxes are more accurate after modeling the local information. Besides, the extra computational overhead of LIE-T2T is minimal. Visualization of detection results of LIE-T2T and T2T are shown in Figure 6. We can see that the detection boxes obtained by our LIE-T2T are very accurate. Every detection box can fit well with the object boundary, yielding outstanding AP_{75}.

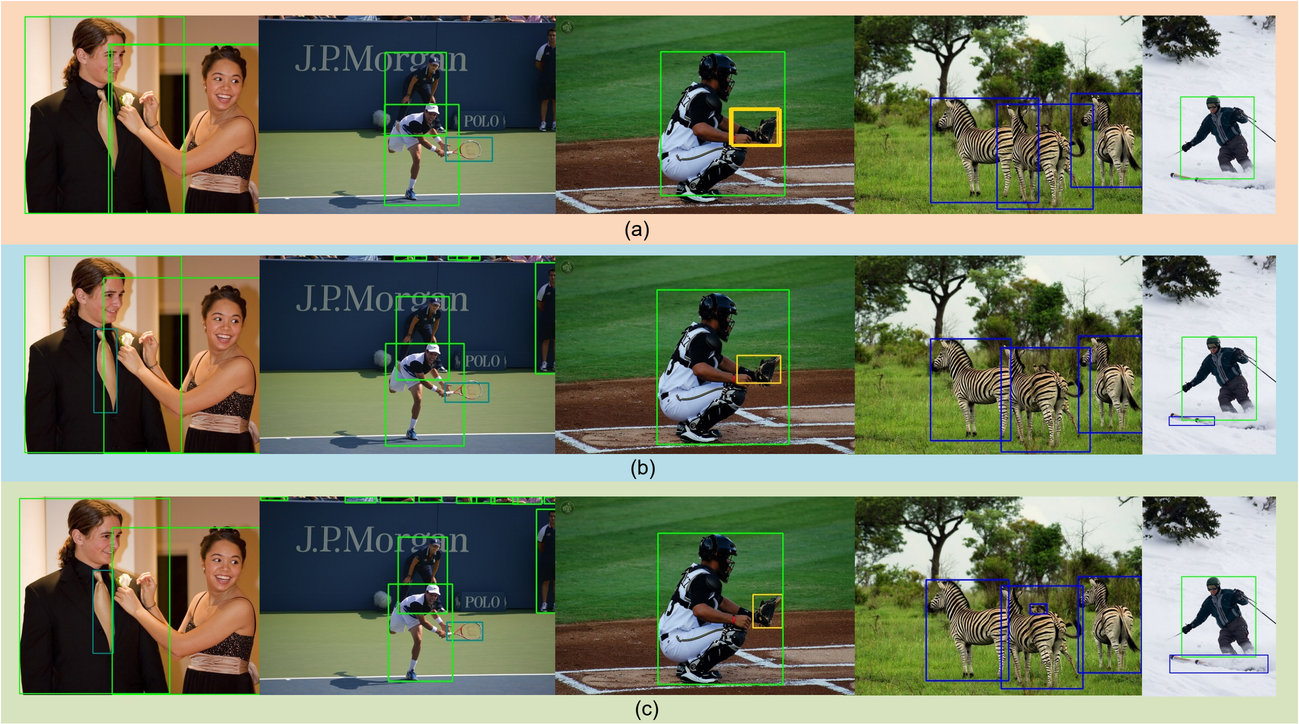

Figure 6. Visualization of detection results. (a) Detection results of directly running object detection with pure-transformer without any modifications. (b) Detection results of object detection using pure-transformer along with the T2T module. (c) Detection results of object detection using pure-transformer along with the LIE-T2T module. We can see that the model, using transformer directly (a), has poor performance for small targets and the regressed bounding boxes are not accurate. After adding the T2T module (b), the detection effect for small targets has been much improved, but the bounding boxes are still not accurate enough. With the addition of our LIE-T2T module (c), the quality of detection results is significant lifted.

Table 4. We change the patch-size and step-size of cutting to evaluate the impact of both the overlap areas and different amounts of information in each token on the detection results.

Patch	Step	Params	FLOPs	FPS	AP	AP_{50}	AP_{75}
8	4	19M	184G	32	41.1	63.3	45.6
16	8	19M	80G	42	40.2	63.2	44.8
24	12	19M	64G	48	38.8	63.0	43.0
32	16	19M	42G	58	36.3	62.5	42.1
8	8	19M	79G	44	39.7	63.2	44.2
16	16	19M	34G	59	35.4	62.3	41.8
24	24	19M	18G	64	34.2	61.5	40.5
32	32	19M	12G	80	33.9	61.0	37.6

Patch and step sizes. We change the patch- and step-size of cutting to evaluate the impact of both the overlap areas and different amounts of information in each token on the detection results. As can be seen in Table 4, the step size has an important relationship with the accuracy of the model. When the step size is too large (e.g., ≥ 16), the model can not effectively output high quality bounding boxes. When the step size is too small (e.g., ≤ 8), there will be an exponential increase in computing overhead. Therefore, it is important to pick the right pair of patch- and step-size.

5. Conclusion

In conclusion, we propose the first pure-transformer detector WB-DETR (DETR-Based Detector without Backbone). The new model is only composed of an encoder and a decoder without any CNN-based backbones. Instead of utilizing a CNN to extract features, WB-DETR serializes the image directly and encodes the local features of input into each individual token. Besides, to allow WB-DETR better make up the deficiency of transformer in modeling local information, we design a LIE-T2T (Local Information Enhancement Tokens-to Token) module to modulate the internal (local) information of each token after unfolding. Unlike other traditional detectors, WB-DETR without backbone is more unify and neat. Experimental results demonstrate that WB-DETR, the first pure-transformer detector without CNN to our knowledge, yields on par accuracy and faster inference speed with only half number of parameters compared with DETR baseline. We encourage researchers to rethink the modules combination (backbone-neck-head) design paradigm for object detection.

References

[1] Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3159–3166, 2019.

[2] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. Monocular 3d pose estimation and tracking by detection. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 623–630. IEEE, 2010.

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018.

[4] 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.

[5] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. Up-detr: Unsupervised pre-training for object detection with transformers. *arXiv preprint arXiv:2011.09094*, 2020.

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

[7] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6569–6578, 2019.

[8] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. Rmpe: Regional multi-person pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2334–2343, 2017.

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

[10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

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

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

[13] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[14] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016.

[15] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *IEEE transactions on pattern analysis and machine intelligence*, 34(7):1409–1422, 2011.

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[17] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6399–6408, 2019.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, 2017.

[19] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 734–750, 2018.

[20] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer, 2014.

[21] 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.

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

[23] 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.

[24] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Advances in neural information processing systems*, pages 2277–2287, 2017.

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

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

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

[28] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with re-

gion proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149, 2017.

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

[30] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in neural information processing systems*, pages 1799–1807, 2014.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[32] Sara Vicente, Joao Carreira, Lourdes Agapito, and Jorge Batista. Reconstructing pascal voc. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 41–48, 2014.

[33] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. *arXiv preprint arXiv:2011.14503*, 2020.

[34] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2 (2019). *URL https://github. com/facebookresearch/detectron2*.

[35] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5987–5995. IEEE Computer Society, 2017.

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

[37] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021.

[38] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z. Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 9756–9765. IEEE, 2020.

[39] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *European Conference on Computer Vision*, pages 474–490. Springer, 2020.

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