Pair DETR: Contrastive Learning Speeds Up DETR Training

Seyed Mehdi Iranmanesh
Amazon
mehdiir@amazon.com

Xiaotong Chen
University of California, Santa Barbara
xchen774@umail.ucsb.edu

Kuo-Chin Lien
Appen
klien@appen.com

Abstract

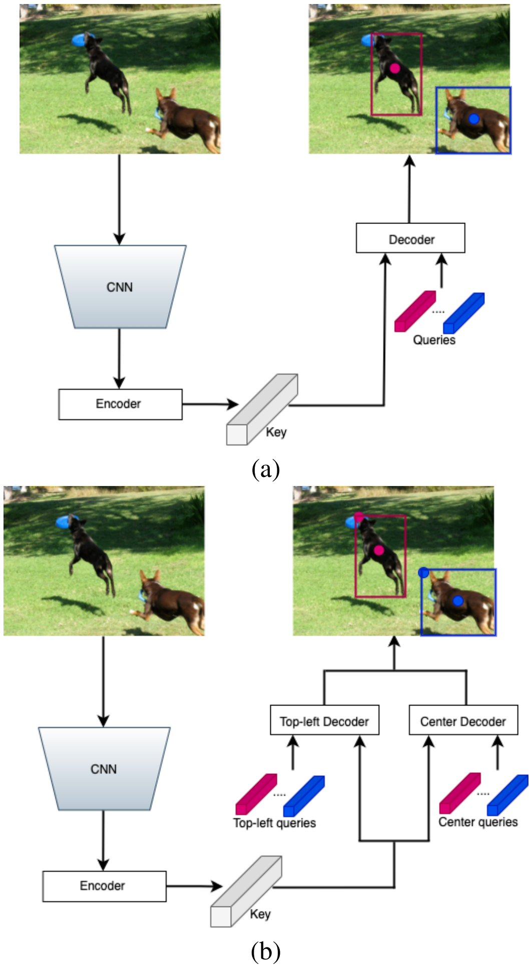
The DETR object detection approach applies the transformer encoder and decoder architecture to detect objects and achieves promising performance. In this paper, we present a simple approach to address the main problem of DETR, the slow convergence, by using representation learning technique. In this approach, we detect an object bounding box as a pair of keypoints, the top-left corner and the center, using two decoders. By detecting objects as paired keypoints, the model builds up a joint classification and pair association on the output queries from two decoders. For the pair association we propose utilizing contrastive self-supervised learning algorithm without requiring specialized architecture. Experimental results on MS COCO dataset show that Pair DETR can converge at least 10x faster than original DETR and 1.5x faster than Conditional DETR during training, while having consistently higher Average Precision scores.

1. Introduction

Object detection utilizing transformers (DETR) [1] is a recent approach that represents object detection as a direct prediction problem via a transformer encoder-decoder [26]. DETR reaches a competitive performance with Faster R-CNN [19] without using hand-designed sample selection and non-maximum suppression (NMS). However, DETR approach suffers from slow convergence on training, which needs large-scale training data and long training epochs to get good performance.

DETR [1] uses a set of learned object queries to reason about the associations of the objects and the global image context to output the final predictions set. However, the learned object query is very hard to explain. It does not have a distinct meaning and the corresponding prediction slots of each object query do not have a unique mode.

Deformable DETR [30], handles the issues of DETR by replacing the global dense attention (self-attention and cross-attention) with deformable attention that attends to a small set of key sampling points and using the high-resolution and multi-scale encoder. This however reduced their AP on the bigger objects which needs global dense attention.



(a)

(b)

Figure 1. Transformer based detection models such as DETR [1] and Deformable DETR [30] rely only on detecting center points (1a). Pair DETR proposes to detect additional point namely Top-left to localize objects more effectively (1b).

Conditional DETR [14] approach is motivated by high dependence on content embeddings and minor contributions made by the spatial embeddings in cross-attention mechanism in DETR. The experimental results in DETR [1] illustrate that if removing the positional embeddings in keys and the object queries from the decoder layer and only using the content embeddings in keys and queries, the detection AP drops. The AP reduction is more observable when the positional embeddings are dropped from the decoder compared the the case when they are dropped from the encoder.

Conditional DETR approach, learns a conditional spatial embedding for each query from the corresponding previous decoder output embedding. It shrinks the spatial range for the content queries to localize the effective regions for class and box prediction. As a result, the dependence on the content embeddings is less and the training is easier.

Inspired by the great success of contrastive learning and self-supervised training in different applications such as CLIP [16] and SimCLR [2], we aim to utilize it in transformer training of DETR.

Two corner points, e.g. the top-left and bottom-right corners, can define the spatial extent of a bounding box, providing an alternative to the typical 4-d descriptor consisting of the box's center point and size which is also used in DETR. This has been used in several bottom-up object detection methods such as [9, 25, 29] and has shown significantly better performance in object localization but not object classification [3].

In order to take advantage of a better localization we propose Pair DETR which consists of a pair of decoders, namely Center decoder and Top-left decoder. Center decoder is responsible to detect the center of bounding boxes while top-left decoder task is to predict top-left of the corresponding bounding boxes (see Fig.1). In addition to the center prediction, the center decoder is responsible for class prediction and the top-left decoder obeys the classification decision. The learned object queries are shared between two decoders, and as a result the order of center and top-left outputs are the same. To have a more discriminative embedding, contrastive learning approach is utilize between these two sets of output. The contributions of the paper are summarized as follows:

- We propose to use contrastive learning on features of two decoders to improve the embedding subspace of decoder and improve the classification performance.
- We incorporate an extra keypoint *Top-left* and associate it with the the standard keypoint – center of a bounding box. This novel two-point Transformer architecture improves the localization performance.
- Extensive experiments are conducted to prove the effectiveness of the proposed model comparing to state-of-the-art on COCO dataset.

2. Related works

2.1. Convolution based object detection

Most object detection models make predictions from carefully designed initial guesses. Anchor boxes or object centers are two main approaches for making the initial guesses. Anchor based solutions which borrow the idea from the proposal based models, can be categorized into two-stage models such as Faster RCNN [19] variants, and one-stage approaches such as SSD [12], and YOLO [17].

Two-stage detectors generate a sparse set of regions of interest (RoIs) and classify each of them by a network. Faster RCNN [19] introduced a region proposal network (RPN), which generates proposals from a set of pre-determined candidate boxes, namely anchor boxes. R-FCN [4] further improves the efficiency of Faster-RCNN by replacing the fully connected sub-detection network with a fully convolutional sub-detection network. DeNet [25] is a two-stage detector which generates RoIs without using anchor boxes. It determines how likely each location belongs to either the top-right, top-left, bottom-right, or bottom-left corner of a bounding box. It then creates RoIs by enumerating all possible corner combinations, and performs the standard two-stage approach to classify each RoI.

One-stage detectors such as SSD and YOLO, remove the RoI pooling step and detects objects in a single network. One-stage detectors are usually more computationally efficient than two-stage detectors while maintaining comparable performance on different challenging benchmarks. SSD directly classifies and refines anchor boxes that are densely defined over feature maps at different scales. YOLOv1 [17] predicts bounding boxes directly from an image, and later is improved in YOLOv2 [18] by utilizing anchors.

Anchor-free models such as YOLOv1, instead of using anchor boxes, predict bounding boxes at points near the center of objects. Only the points near the center are used since they are considered to be able to produce higher quality detection. YOLOv1 struggles with low recall, and thus YOLOv2 chooses to go back employing anchor boxes. Compared to YOLOv1, FCOS [24] can utilize all points in a ground truth bounding box to predict the bounding boxes and make comparable results to anchor based solutions. CenterNet [28] predicts the center, width, and height of objects with hourglass networks, demonstrating promising performance. CornerNet [9] is another keypoint-based approach, which directly detects an object using a pair of corners and groups them to form the final bounding box.

2.2. Transformer based object detection

DETR successfully applies Transformers to object detection, effectively removing the need for many hand-designed components like non-maximum suppression or anchor generation. The high computation complexity issue, caused by

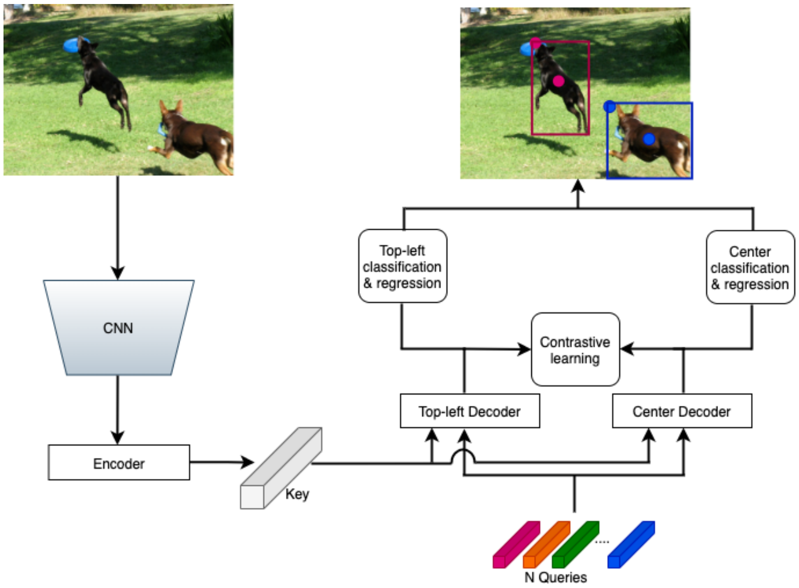


Figure 2. **Pair DETR** contains additional decoder comparing to the other DETR based detection models. One decoder is dedicated to detection of center points (Center decoder) and the other is dedicated to locating top-left points (Top-left decoder). Pair DETR is optimized jointly using classification and regression losses for each decoder and contrastive loss between the two mentioned decoders.

the global encoder self attention, is handled by sparse attentions in Deformable DETR [30] or by utilizing adaptive clustering transformer [27]. Deformable DETR [30] is introduced to restrict each object query to attend to a small set of key sampling points around the reference points, instead of all points in feature map. Using multi-scaling and utilizing high resolution feature maps, it can mitigate potential losses in the detection of small objects.

The other critical issue, slow training convergence in DETR [1], has been attracting a lot of recent research attention. Deformable DETR [30] adopts deformable attention to replace decoder cross-attention. TSP approach [22] combines the FCOS and R-CNN-like detection heads and removes the cross-attention modules. With unsupervised pretraining of the transformers, UP-DETR [5] enhances DETR performance on object detection precision and also speed of model convergence. Conditional DETR [14] encodes the reference point as the query position embedding. It utilizes the reference point to generate position embedding as the conditional spatial embedding in the cross-attention layers. Using spatial queries shrinks the spatial range for the content queries to localize the regions for box and class prediction. Therefore, the dependence on the content embeddings is relaxed and the training convergence is faster. Our proposed approach is based on conditional DETR to improve the performance of DETR object detection while taking advantage of conditional DETR architecture for faster training convergence.

3. Methodology

In this section, we first introduce preliminaries and then discuss details of our method.

3.1. Contrastive learning

Contrastive learning has become popular due to leading state-of-the-art results in different unsupervised representation learning tasks. The goal is to learn representations by contrasting positive pairs against negative pairs [7] and is used in various computer vision tasks, including data augmentation [15], or diverse scene generation [23]. The main idea of contrastive learning is to move the similar pairs near and dissimilar pairs far in the embedding subspace.

In this work, we follow a similar approach to SimCLR framework [2] for contrastive learning. SimCLR consists of four main components: a stochastic data augmentation method which generates positive pairs, an encoding network (f) that extracts the embedding representation vectors from augmented samples, a small projector head (g) that maps representations to the loss space, and a contrastive loss function that enforces more discriminative embedding representation between positive and negative pairs. Given a random mini-batch of N samples, SimCLR generates N positive pairs using the data augmentation method. For all positive pairs, the remaining 2N-1 augmented samples are treated as negative examples. Let $\mathrm{h_i} = \mathrm{f}(\mathrm{x_i})$ be the representations of all 2N samples and $\mathrm{z_i} = \mathrm{g}(\mathrm{h_i})$ be the projections of these representations. Let $\mathrm{sim}(\mathrm{u}, \mathrm{v}) = \mathrm{u}^{\mathrm{T}}\mathrm{v}/||\mathrm{u}||||\mathrm{v}||$ denote the dot product between

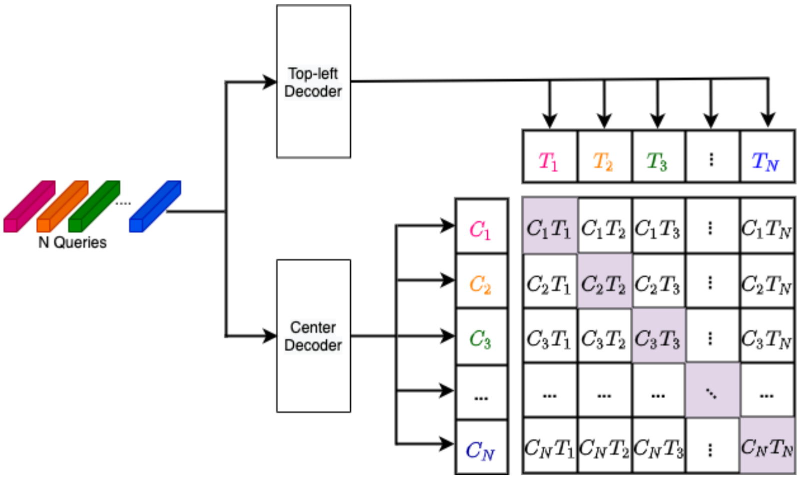


Figure 3. Pair DETR utilizes contrastive learning to maximize the similarity between corresponding output embeddings (positive) of Center and Top-left decoders and minimizing the similarity to other output embeddings (negatives).

ℓ_2 normalized u and v. Then the loss function for a positive pair of examples (i, j) is defined as:

$$\ell_{i,j} = -\log \frac{\exp(\mathrm{sim}(\mathrm{z_i}, \mathrm{z_j})/\tau)}{\Sigma_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp(\mathrm{sim}(\mathrm{z_i}, \mathrm{z_j})/\tau)}, \tag{1}$$

where $\mathbb{1} \in \{0, 1\}$ is an indicator function evaluating to 1 iff $\{\mathrm{k} \neq \mathrm{i}\}$ and τ denotes a temperature parameter. The final loss which is also known as *NT-Xent* [15, 21] considers the average of losses computed across all positive pairs, both (i, j) and (j, i), in a mini-batch [2].

3.2. DETR

The proposed approach follows detection transformer (DETR) and predicts all the objects at once without the need for anchor generation or NMS. DETR architecture consists of a CNN backbone, a transformer encoder, a transformer decoder, and object class and box position predictors. The goal of transformer encoder is to enhance the embeddings output from the CNN backbone. It is a stack of multiple encoder layers, where each layer mainly consists of a self-attention layer and a feed-forward layer.

The transformer decoder is a stack of decoder layers. Each decoder layer, consists of three main components which are self-attention, cross-attention, and feed-forward layers, respectively. Self-attention layer performs interactions between the embeddings, outputted from the previous decoder layer and is utilized to remove duplication in prediction. Cross-attention layer combines the embeddings output from the encoder to refine the decoder embeddings for enhancing class and box prediction. Conditional DETR utilizes conditional cross-attention mechanism and forms the query by concatenating the content query $\mathrm{c_q}$, outputting from decoder self-attention, and the spatial query $\mathrm{p_q}$.

3.3. Pair DETR

The proposed Pair DETR mechanism forms by creating a pair of points. There are evidences which show the superiority of using two corners (e.g. the top-left corner and bottom-right corner) in terms of localization comparing to the other forms of object detection models which using center, width, height of bounding box representation. However, corner based models in general perform worse for the classification part [3]. One possible reason might be due to the fact that the corners contain less contextual information of the object inside the bounding box comparing to the center. Pair DETR is to taking advantage of both of these models by considering a pair of the center of bounding box and one corner of the same bounding box. Having only one corner in addition to the center is enough since the bottom-right corner is reciprocal to the top-left one. In addition, to associate the corresponding points belonging to the same bounding box, the center and one corner probably have more similar contents comparing to two opposite corners.

Two parallel decoders are employed in Pair DETR. Feature maps generated from the encoder are used as common keys by the two decoders. Two decoders are designed to detect top-left and center points, respectively. Each decoder has its own classification and regression loss and both decoders are trained jointly. The same set of learnt positional encodings that are referred to as object queries are fed to the pair of decoders (i.e, top-left decoder and center decoder). Specifically, the first object query from this set which is fed to the top-left decoder will be specialized in detecting the top-left of first bounding box. In addition, this object query is fed to the center decoder to detect the center of corresponding bounding box. These two embeddings from the two decoders are constructing the positive pair. Contrastive learning is utilized to keep consistent representation from

Table 1. Comparison between Pair DETR and previous DETR variants on COCO 2017 val. Our Pair DETR approach for all the backbones including high resolution (DC5-R50 and DC5-R101) and low resolution (R50 and R101) is at least 10× faster than the original DETR. In addition, Pair DETR performs almost 1.5× faster than conditional DETR. Pair DETR is empirically superior to other two single-scale DETR variants.

Model	#epochs	AP	AP$_{50}$	A$_{75}$	AP$_S$	AP$_M$	AP$_L$
DETR-R50	500	42.0	62.4	44.2	20.5	45.8	61.1
DETR-R50	50	34.9	55.5	36.0	14.4	37.2	54.5
Conditional DETR-R50	50	40.9	61.8	43.3	20.8	44.6	59.2
Conditional DETR-R50	75	42.1	62.9	44.8	21.6	45.4	60.2
Conditional DETR-R50	108	43.0	64.0	45.7	22.7	46.7	61.5
Pair DETR-R50	50	42.2	63.4	44.9	21.5	45.5	61.1
Pair DETR-R50	75	43.0	64.6	45.7	22.0	46.2	61.5
DETR-DC5-R50	500	43.3	63.1	45.9	22.5	47.3	61.1
DETR-DC5-R50	50	36.7	57.6	38.2	15.4	39.8	56.3
Conditional DETR-DC5-R50	50	43.8	64.4	46.7	24.0	47.6	60.7
Conditional DETR-DC5-R50	75	44.5	65.2	47.3	24.4	48.1	62.1
Conditional DETR-DC5-R50	108	45.1	65.4	48.5	25.3	49.0	62.2
Pair DETR-DC5-R50	50	44.6	65.4	47.5	24.5	48.1	61.9
Pair DETR-DC5-R50	75	45.5	66.0	47.9	24.8	48.5	62.4
DETR-R101	500	43.5	63.8	46.4	21.9	48.0	61.8
DETR-R101	50	36.9	57.8	38.6	15.5	40.6	55.6
Conditional DETR-R101	50	42.8	63.7	46.0	21.7	46.6	60.9
Conditional DETR-R101	75	43.7	64.9	46.8	23.3	48.0	61.7
Conditional DETR-R101	108	44.5	65.6	47.5	23.6	48.4	63.6
Pair DETR-R101	50	43.6	64.8	46.7	22.4	47.2	62.0
Pair DETR-R101	75	44.4	65.8	47.4	24.3	48.4	62.7
DETR-DC5-R101	500	44.9	64.7	47.7	23.7	49.5	62.3
DETR-DC5-R101	50	38.6	59.7	40.7	17.2	42.2	57.4
Conditional DETR-DC5-R101	50	45.0	65.5	48.4	26.1	48.9	62.8
Conditional DETR-DC5-R101	75	45.6	66.5	48.8	25.5	49.7	63.3
Conditional DETR-DC5-R101	108	45.9	66.8	49.5	27.2	50.3	63.3
Pair DETR-DC5-R101	50	45.7	66.4	48.9	26.4	49.3	63.6
Pair DETR-DC5-R101	75	46.1	67.2	49.3	26.7	49.8	63.9
Other Single-scale DETR variants							
Deformable DETR-R50-SS	50	39.4	59.6	42.3	20.6	43.0	55.5
UP-DETR-R50	150	40.5	60.8	42.6	19.0	44.4	60.0
UP-DETR-R50	300	42.8	63.0	45.3	20.8	47.1	61.7
Deformable DETR-DC5-R50-SS	50	41.5	61.8	44.9	24.1	45.3	56.0

outputs of two decoders. Given 2N queries and a positive pair, similar to [2, 16], we treat one pair as positive and the other 2N-1 queries as negative examples. The procedure is depicted in Figures 2 and 3.

4. Results

4.1. Dataset

Same as other work in the DETR family, we conduct experiments on COCO 2017 dataset [11]. Our models are trained on the train set, and evaluated on the val set.

4.2. Implementation details

Pair DETR is based on conditional DETR architecture [14] and consists of the CNN backbone, transformer encoder, two transformer decoders, prediction feed-forward networks (FFNs) following each decoder layer (the last decoder layer and the 5 internal decoder layers) with parameters shared among the 6 prediction FFNs. Classification network (FFN) is shared between the two decoders (center and top-left decoders) while the regression network is separated. The hyperparameters are the same as DETR and conditional DETR. Conditional spatial embeddings are employed as the spatial queries for conditional multi-head cross-attention

Table 2. A comparison of Pair DETR to multi-scale and higher-resolution DETR variants. Pair DETR is a single-scale approach. However our approach when utilizing high resolution backbone such as DC5-R50 and DC5-R101 outperforms the two multi-scale and higher-resolution DETR variants. This is with using only single scale and without using high resolution encoder.

Model	#epochs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster RCNN-FPN-R50 [19]	36	40.2	61.0	43.8	24.2	43.5	52.0
Faster RCNN-FPN-R50	108	42.0	62.1	45.5	26.6	45.5	53.4
Deformable DETR-R50	50	43.8	62.6	47.7	26.4	47.1	58.0
TSP-FCOS-R50 [22]	36	43.1	62.3	47.0	26.6	46.8	55.9
TSP-RCNN-R50	36	43.8	63.3	48.3	28.6	46.9	55.7
TSP-RCNN-R50	96	45.0	64.5	49.6	29.7	47.7	58.0
Pair DETR-DC5-R50	50	44.6	65.4	47.5	24.5	48.1	61.9
Pair DETR-DC5-R50	108	45.8	66.2	48.8	25.7	49.0	63.0
Faster RCNN-FPN-R101	36	42.0	62.5	45.9	25.2	45.6	54.6
Faster RCNN-FPN-R101	108	44.0	63.9	47.8	27.2	48.1	56.0
TSP-FCOS-R101	36	44.4	63.8	48.2	27.7	48.6	57.3
TSP-RCNN-R101	36	44.8	63.8	49.2	29.0	47.9	57.1
TSP-RCNN-R101	96	46.5	66.0	51.2	29.9	49.7	59.2
Pair DETR-DC5-R101	50	45.7	66.4	48.9	26.4	49.3	63.6
Pair DETR-DC5-R101	108	46.6	67.5	49.8	27.4	50.6	64.4

and the combination of the spatial query (key) and the content query (key) is done through concatenation other than addition. In the first cross-attention layer there are no decoder content embeddings, we make simple changes based on the DETR implementation: concatenate the positional embedding predicted from the object query (the positional embedding) into the original query (key).

The prediction unit for spatial queries is an FFN and consists of learnable linear projection + ReLU + learnable linear projection: $\mathrm{s} = \mathrm{FFN}(\mathrm{o_q})$. The 2D coordinates are normalized by the sigmoid function to use for forming conditional spatial query.

We utilized the Hungarian algorithm to find an optimal bipartite matching between predicted and the ground-truth object following DETR [1] and then form the loss function for computing and back-propagate the gradients. We follow Deformable DETR [30] to formulate the loss: the same loss function with 300 object queries, the same matching cost function, and the same trade-off parameters; The classification loss function is focal loss [10], and the box regression loss (including L1 and GIoU [20] loss) is the same as DETR [1]. Each decoder has its own classification and regression loss. Contrastive loss is also utilized between two decoders.

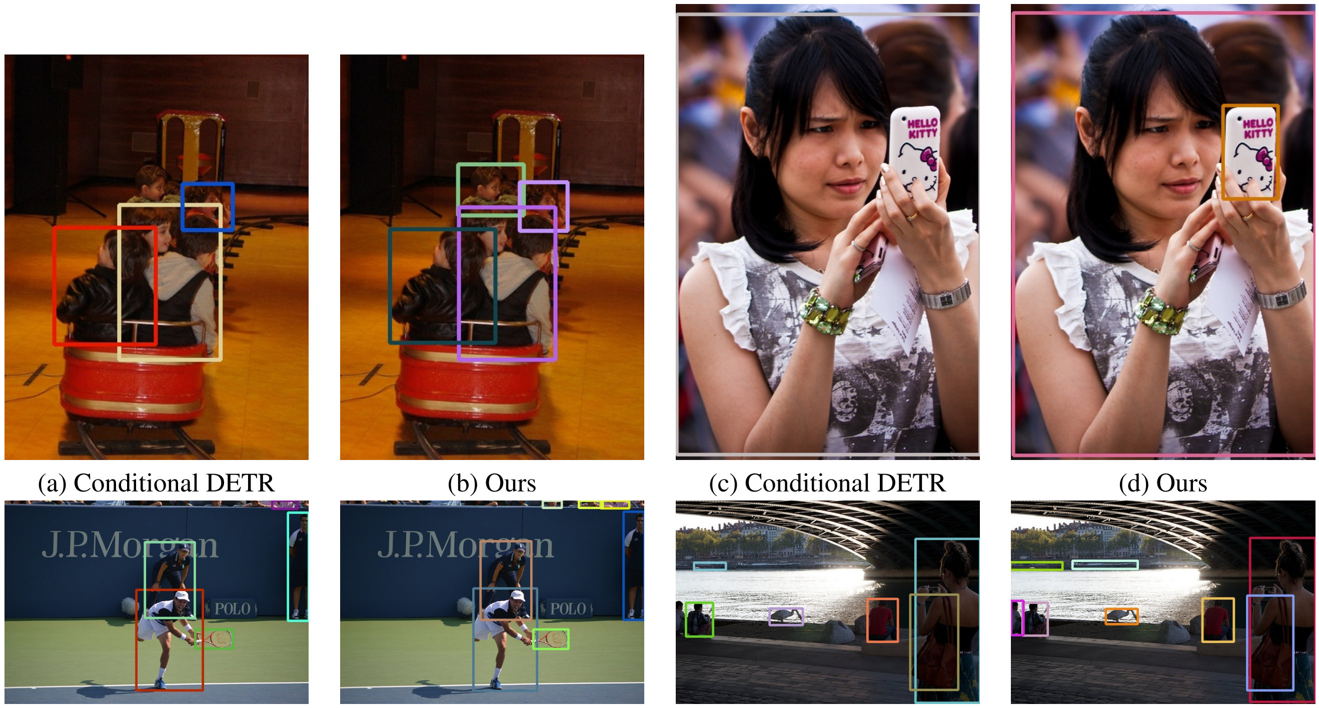
Training is performed following the DETR training protocol [1]. The backbone is the pretrained model on ImageNet from torchvision with batchnorm layers fixed, and the transformer parameters are initialized using the Xavier initialization scheme [6]. The weight decay is set to be 10^{-4}. The AdamW [13] optimizer is employed. The learning rates for the backbone and the transformer are initially set to be 10^{-5} and 10^{-4}, respectively. The dropout rate in transformer is 0.1. The learning rate is dropped by a factor of 10 after 40 epochs for 50 training epochs, after 60 epochs for 75 training epochs, and after 80 epochs for 108 training epochs. We use data augmentation following DETR scheme: resizing the input images whose shorter side is between 480 to 800 pixels while the longer side is by at most 1333 pixels; randomly crop the image such that a training image is cropped with probability 0.5 to a random rectangular patch. We use the standard COCO to evaluate and report the average precision (AP), and the AP scores at 0.50, 0.75 and for the small, medium, and large objects.

4.3. Comparison to DETR

We compare the proposed Pair DETR to the original DETR [1] and conditional DETR [14]. We follow [1] and report the results over four backbones: ResNet-50 [8], ResNet-101, and their high resolution extensions DC5-ResNet-50 and DC5-ResNet-101.

The corresponding DETR models are named as DETR-R50, DETR-R101, DETR-DC5-R50, and DETR-DC5-R101, respectively. Our models are named as Pair DETR-R50, Pair DETR-R101, Pair DETR-DC5-R50, and Pair DETR-DC5-R101, respectively.

Table 1 presents the results from DETR, conditional DETR, and Pair DETR. DETR with 50 training epochs performs much worse than 500 training epochs. Conditional DETR with 50 training epochs for R50 and R101 as the backbones performs slightly worse than DETR with 500 training epochs. However Pair DETR (backbone R50 and R101) performs slightly better than DETR with 500 training epochs. In addition, Pair DETR with 50 training epochs for DC5-R50 and DC5-R101 could outperform DETR with 500



(a) Conditional DETR (b) Ours (c) Conditional DETR (d) Ours

(e) Conditional DETR (f) Ours (g) Conditional DETR (h) Ours

Figure 4. Visual comparison on COCO validation set. Comparing to conditional DETR [14], Pair DETR shows superior performance in detecting objects blending in the background (4a vs. b, 4g vs. h), locating smaller objects contained within larger objects (4c vs. d), as well as detecting objects around the boundaries of images (4e vs. f, 4g vs. h).

training epochs by a bigger margin. Pair DETR for the three backbones with 50 training epochs performs slightly better than conditional DETR with 75 training epochs (the only exception is backbone R101). Figure 4 visualizes the detection performance of Pair vs. conditional DETR on COCO dataset with 50 training epochs. In addition, Pair DETR with 75 training epochs outperform conditional DETR with 75 training epochs with a larger margin and for all the four backbones. In summary, training Pair DETR for all the backbones is at least 10× faster than the original DETR. In addition, training Pair DETR for all the backbones is 1.5× faster than corresponding conditional DETR.

In addition, we report the results of single-scale DETR extensions: UP-DETR [5] and deformable DETR-SS [30] in Table 1. Our results over R50 and DC5-R50 are better than deformable DETR-SS: 42.2 vs. 39.4 and 44.6 vs. 41.5. The comparison might not be fully fair as for example parameter and computation complexities are more different in these extensions, but it implies that the conditional cross-attention mechanism is beneficial. Compared to UP-DETR-R50, our results with fewer training epochs are obviously better.

Overall, some people may take the higher AP in Table 1 as granted as Pair DETR obviously has more parameters than some of the other methods, e.g., the original DETR and the Conditional DETR, to fit the data. However, in addition to accuracy, it is important to read it from another perspective: Pair DETR demonstrates significantly faster convergence than its counterparts, although having a larger size of model.

4.4. Comparison to multi-scale and higher-resolution DETR variants

It has been shown in prior work that conventional wisdom in computer vision such as embracing higher resolution and multi scale can also boost object detection accuracy for DETRs. As they are techniques orthogonal to this

Table 3. A comparison of different settings for Pair DETR. We investigate the effect of adding each component to the proposed Pair DETR. Addition of contrastive learning showed effective improvement. In addition, localization of bounding boxes based on two points is more accurate compared to regressing it directly from the center.

Model	Cont. learning	w, h	Pairs coordinates	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Pair DETR-R50 (a)		✓		41.1	62.1	43.5	20.9	44.8	59.3
Pair DETR-R50 (b)	✓	✓		41.8	63.3	44.1	21.1	45.2	60.7
Pair DETR-R50 (c)	✓		✓	42.1	63.6	44.5	21.3	45.4	61.0
Pair DETR-R50 (d)	✓	✓	✓	42.2	63.8	44.5	21.5	45.5	61.1

Table 4. A comparison of different matching strategies for Pair DETR.

Model	head sep.	head sh.	Tl cl. cost	Cent. cl. cost	AP	AP_{50}	A_{75}	AP_S	AP_M	AP_L
Pair DETR-R50 (1)	✓		✓	✓	41.4	62.5	43.8	21.0	44.9	59.8
Pair DETR-R50 (2)		✓	✓	✓	42.0	63.4	44.2	21.3	45.3	60.7
Pair DETR-R50 (3)		✓		✓	42.2	63.8	44.5	21.5	45.5	61.1

paper, it is possible for future extension of Pair DETR also benefits from multi-scale attention and/or higher-resolution encoder. Although we did not perform these implementations, Table 2 lists the performance of these methods and Pair DETR using higher resolution backbones as a comparison. It can be seen that our methods are on par or better than these recent work.

4.5. Ablation

Effect of contrastive loss and regressing bounding box. We consider four different models to study the effect of different components of the Pair DETR. All models are trained for 50 epochs and the classification head is shared between two decoders. However, each decoder has it own regression head. In the setting (a) contrastive learning is not employed and also (w, h) is used to regress the bounding box from its center. This model performance is very similar to the conditional DETR.

In second setting (b), the model is jointly trained using contrastive loss and the classification and regression loss functions. Adding contrastive loss function causes the decoders embedding subspace to be more discriminative and consequently the detection performance is improved. In the third setting (c), we utilize the coordinates of two points to measure width and height instead of regressing them directly from the center points. Therefore the conversion function becomes:

$$\mathrm{T(w,h)} = \Big((\mathrm{x_{tl}} - \mathrm{x_c}) \times 2, (\mathrm{y_{tl}} - \mathrm{y_c}) \times 2\Big), \qquad (2)$$

where $(\mathrm{x_{tl}}, \mathrm{y_{tl}})$ and $(\mathrm{x_c}, \mathrm{y_c})$ are the coordinates of top-left and center points, respectively. This conversion makes the model to consider two important points instead of relying on one point (center) and it has a positive impact on the detection performance. In the last setting (d), width and height are extracted using Eq.2 and also predicted independently from center decoder. The conversion function for width and height is taking average of them as follows:

$$\mathrm{T(w,h)} = \Big(\frac{(\mathrm{x_{tl}} - \mathrm{x_c}) \times 2 + \mathrm{w_c}}{2}, \frac{(\mathrm{y_{tl}} - \mathrm{y_c}) \times 2 + \mathrm{h_c}}{2}\Big), \qquad (3)$$

where $(\mathrm{x_{tl}}, \mathrm{y_{tl}})$ and $(\mathrm{x_c}, \mathrm{y_c})$ are the coordinates of top-left and center points, respectively. $(\mathrm{w_c}, \mathrm{h_c})$ is the predicted width and height from the center decoder. This setting outperforms the other settings as it is presented in Table 3. Here we take the average of the calculated (w,h) from the pair coordinates as well as the predicted width and height from the center decoder.

Strategies for Hungarian matching. We evaluate different strategies for Hungarian matching. As it is explained in original DETR paper, after the bounding box prediction step, they are selected in a way to minimize the classification and regression costs. In this experiment, we are interested to explore the effect of classification cost on Hungarian matching and model performance. Three different settings are considered and in all of them the model is trained with 50 training epochs. In the first setting (1), each decoder has its own classification head. Moreover, Top-left decoder has the same role as center decoder in classification cost of Hungarian matching. This model performs worst among all the three settings. In the second setting (2), the classification head is shared between two decoders, but still the classification cost of Top-left decoder plays equal role and is considered in matching. In the third setting (3), the classification head is shared and only the classification cost of center decoder is considered for matching. As it is shown in Table 4, this model outperforms the other two settings. One possible reason for this can be due to the fact that the center points contain more representative features related to the content (class id) of the bounding box compared to extreme points or corner points. Therefore this setting is selected for our proposed Pair DETR model.

5. Conclusions

In this work, we introduce a simple approach for considering extra point in DETR which detects objects using a pair, including one center keypoint and one corner (Top-left). For this purpose, one additional decoder is employed. Using this approach, we leverage more than one keypoint for the localization of a bounding box. In addition, we utilize a contrastive learning approach as an auxiliary task to make the output embeddings of two decoders to be more discriminative. Although the presented implementation is based upon conditional DETR, the simplicity of the idea allows it easily extended to other DETR family detectors e.g., Deformable DETR [30] in the future, expectably boosting the performance of the base detector by a considerable margin.

References

[1] 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. 1, 2, 3, 6

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 2, 3, 4, 5

[3] Yihong Chen, Zheng Zhang, Yue Cao, Liwei Wang, Stephen Lin, and Han Hu. Reppoints v2: Verification meets regression for object detection. *Advances in Neural Information Processing Systems*, 33, 2020. 2

[4] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016. 2

[5] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. Up-detr: Unsupervised pre-training for object detection with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1601–1610, 2021. 3, 7

[6] 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. 6

[7] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006. 3

[8] 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. 6

[9] 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. 2

[10] 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. 6

[11] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 5

[12] 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. 2

[13] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. 2018. 6

[14] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3651–3660, 2021. 1, 3, 5, 6, 7

[15] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 3, 4

[16] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021. 2, 5

[17] 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. 2

[18] 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. 2

[19] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015. 1, 2, 6

[20] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 658–666, 2019. 6

[21] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pages 1857–1865, 2016. 4

[22] Zhiqing Sun, Shengcao Cao, Yiming Yang, and Kris M Kitani. Rethinking transformer-based set prediction for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3611–3620, 2021. 3, 6

[23] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020:*

16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16, pages 776–794. Springer, 2020. 3

[24] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9627–9636, 2019. 2

[25] Lachlan Tychsen-Smith and Lars Petersson. Denet: Scalable real-time object detection with directed sparse sampling. In *Proceedings of the IEEE international conference on computer vision*, pages 428–436, 2017. 2

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 1

[27] Minghang Zheng, Peng Gao, Xiaogang Wang, Hongsheng Li, and Hao Dong. End-to-end object detection with adaptive clustering transformer. *arXiv preprint arXiv:2011.09315*, 2020. 3

[28] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 2

[29] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krahenbuhl. Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 850–859, 2019. 2

[30] 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. 1, 3, 6, 7, 9