Supplementary Material SeqCo-DETR: Sequence Consistency Training for Self-supervised Object Detection with Transformers

Guoqiang Jin[1]
jinguoqiang@sensetime.com
Fan Yang[2, 3]
yangfan_2022@ia.ac.cn
Mingshan Sun[1]
sunmingshan@sensetime.com
Ruyi Zhao[1]
zhaoruyi@sensetime.com
Yakun Liu[1]
liuyakun1@sensetime.com
Wei Li[1]
liwei1@sensetime.com
Tianpeng Bao[1]
baotianpeng@sensetime.com
Liwei Wu[1]
wuliwei@sensetime.com
Xingyu Zeng[1]
zengxingyu@sensetime.com
Rui Zhao[1,4]
zhaorui@sensetime.com

[1] SenseTime Research
[2] Institute of Automation, CAS
[3] Peng Cheng Lab
[4] Qing Yuan Research Institute, Shanghai Jiao Tong University, Shanghai, China

1 Overview

We organize the supplementary material as follows. The implementation details are given in Sec. 2. More results and the ablation study are presented in Sec. 3. Then, we visualize the matched proposal pairs between the two branches in Sec. 4. Finally, we discuss the limitations and future work of our approach in Sec. 5.

2 Implementation Details

Training details. Our training procedure consists of two stages: pre-training and fine-tuning. In the pre-training stage, we add an additional feedforward network (FFN) head for projecting sequence features. The FFN comprises 2 hidden layers, and the number of hidden layer neurons is 256. Following DETReg [1], Selective Search [12], based on OpenCV [2], is used to generate the initial foreground proposals. During pre-training, the classification head has two categories: foreground and background. We only use the top 30 proposals from Selective Search in one image. The backbone of the proposed method is ResNet-50 [9] and is initialized by SwAV [3]. In the pre-training stage, for the ImageNet100 (IN100) [5] dataset, the number of epochs is 50, the batch size is 24, and the initial learning rate is $2 \cdot 10^{-4}$, which is decayed after 40 epochs by a factor of 10. For the ImageNet (IN1K) [6] dataset, the pre-training epoch is 5. The parameters in the fine-tuning stage are also the same as DETReg. On MS COCO [10], the number of epochs is 50, the batch size is 4, and the initial learning rate is $2 \cdot 10^{-4}$, which is decayed after 40 epochs by a factor of 10. On PASCAL VOC [7], the number of epochs is 100, the batch size is 4, and the initial learning rate is $2 \cdot 10^{-4}$, which is decayed after 70 epochs by a factor of 10. For the few-shot object detection task, we follow the standard protocol [13] as used in DETReg. We pre-train all the ablation studies on IN100 and fine-tune on COCO. Experiments are carried out on 8 * NVIDIA V100 GPUs.

Algorithm pseudocode. The algorithm flow of the proposed method is summarized in Algorithm 1.

3 More results

3.1 Results on few-shot object detection task

Following the standard procedure in the few-shot setting [1, 13], we split the COCO dataset into 60 base classes and 20 novel classes. The 60 base classes contain the full data, while each novel class has only $k \in 10, 30$ samples. We first fine-tune the model on the base classes using the default parameters on COCO. Then, we fine-tune it on the base and novel classes, the same as in DETReg. Specifically, for $k = 10$, we fine-tune 30 epochs with learning rate of $2 \cdot 10^{-5}$, for $k = 30$, we fine-tune 50 epochs with learning rate of $4 \cdot 10^{-5}$. The results are reported on the novel classes. As listed in Tab. 1, our SeqCo-DETR outperforms DETReg by 2.0 and 0.9 points in $k = 10$ and $k = 30$ settings, respectively, proving our method has better feature representation capabilities and is more feasible for various object detection tasks. With better feature representation ability, good performance can be obtained even with only a small amount of data during fine-tuning.

3.2 More ablation results

Self-supervised loss strategy. As listed in Tab. 2, we compare the effects of different self-supervised losses. Predictor + BYOL loss follows the BYOL [8] architecture, which adds an extra MLP layer on the online branch to make the two branches asymmetrical and the inputs of online and momentum branches are also exchanged to create symmetry loss. MoCo loss comes from MoCo v3 [4] symmetric version. Interestingly, L1 and L2 loss can achieve satisfactory results, thus, the L2 loss is adopted in the experiments.

Algorithm 1 Pseudo code of SeqCo-DETR in a PyTorch-like style.

```
# model_momentum: momentum backbone + encoder + decoder + head
# model_online: online backbone + encoder + decoder + head
# mse: mean squared error, i.e., L2 loss
# augment: image augmentation
# matcher: bipartite matching algorithm
# rps: region proposals from Selective Search
# criterion: classification and regression loss

for param in model_momentum.parameters():
   param.requires_grad = False

def similarity_loss(e1, e2, ids_ssl):
   loss = mse(e1-e2[ids_ssl]).sum().mean()
   return loss

for x in dataloader: # load a batch x with B samples
   x1, x2 = weak_augment(x), strong_augment(x)

   with torch.no_grad():
      cls_m, box_m, prj_m = model_momentum(x1)

   cls_o, box_o, prj_o = model_online(x2)

   idx_rps = matcher((cls_o, box_o), rps)
   loss_rps = criterion((cls_o, box_o), rps, idx_rps)

   idx_ssl = matcher((cls_m, box_m), (cls_o, box_o))
   loss_ssl = similarity_loss(prj_m, prj_o, idx_ssl)

   loss = loss_rps + loss_ssl
   loss.backward()

   model_online.update()
   model_momentum.update()
```

Model	Novel AP 10	Novel AP 30	Novel AP_{75} 10	Novel AP_{75} 30
FRCN+ft-full [13]	9.2	12.5	9.2	12.0
Deformable DETR (Supervised CNN)‡	23.3	28.4	25.4	31.7
UP-DETR (Deformable DETR)†	23.9	27.1	26.3	29.4
DETReg w/o feature embedding†	24.2	26.1	26.5	28.2
DETReg [1]	25.0	30.0	27.6	**33.7**
SeqCo-DETR	**27.0**	**30.9**	**29.7**	33.4

Table 1: Comparison results on few-shot detection task on COCO, evaluated on the novel classes. †: We run the method on our codebase. ‡: Results are provided by DETReg [1].

Model	Self-supervised loss strategy	AP
SeqCo-DETR	L1 loss	45.6
	L2 loss	**45.8**
	MoCo loss	45.4
	Predictor + BYOL loss	45.2

Table 2: Comparison of self-supervised loss strategies on MS COCO `val2017`.

Training efficiency. As listed in Tab. 3, our SeqCo-DETR achieves better results on COCO and VOC compared with DETReg [1] and JoinDet [14]. More details are in the main paper. SeqCo-DETR's training time is not significantly longer than DETReg's, and it uses only half the number of V100 GPUs as JoinDet. The proposed mask strategy has minimal impact on training complexity. It is worth noting that even slight improvement in downstream detection tasks with the large COCO object detection dataset can prove the effectiveness of the pre-training method, since the pre-training only modifies the object detection network's weights without changing the object detection algorithm itself. Our SeqCo-DETR achieves significant improvements of 0.3 AP and 0.6 AP on COCO and VOC, respectively, compared with DETReg. These improvements demonstrate the efficacy of the self-supervised pre-training for object detection.

Model	AP@COCO	AP@VOC	Pre-training time (h)	No. V100 GPUS
DETReg [1]	45.5	63.5	24	8
JoinDet [14]	45.6	63.7	28	16
SeqCo-DETR (w/o mask)	45.6	63.9	27	8
SeqCo-DETR	**45.8**	**64.1**	28	8

Table 3: Comparison results of different methods.

3.3 More results on the mask strategy

As mentioned in the main paper, we propose a complementary mask strategy for the self-supervised learning method for object detection. To determine the optimal combinations and parameters of the mask strategy, we conduct multiple ablation experiments, which are detailed below.

3.3.1 Single mask strategy for the online branch

In this section, we conduct several experiments to determine the optimal proportion of the single image mask. As shown in Fig. 1, we demonstrate the images after being masked with different proportions. Specifically, we only add the mask to the online branch, and the patch size of the mask is 16. Experiments on the patch size are discussed in Sec. 3.3.3.

(a) 10% (b) 30% (c) 50% (d) 70% (e) 90% (f) 99%

Figure 1: Illustration of different proportions of masked images.

The results are listed in Tab. 4, where `Rnd(30,80)` stands for the mask proportions are randomly sampled from 30% to 80%. From the table, our results remain stable at first and then decrease as the mask proportion increases. We also try to use the random proportion of mask during pre-training, and the `Rnd(30,80)` achieves the best result. In contrast, the performance of our baseline DETReg [1] decreases when adding masks to the input image. Specifically, when there is no mask added to the image, i.e., the mask proportion is 0, the

Mask proportion(%)	0	30	50	70	90	99	Rnd(30,80)	Rnd(0,99)
DETReg†	45.4	-	45.0	-	-	-	45.3	-
SeqCo-DETR	**45.6**	**45.6**	**45.6**	45.5	45.3	45.0	**45.7**	45.3

Table 4: Comparison of different proportions of single masks added to the online branch, evaluated on MS COCO `val2017`. †: We run the method on our codebase.

Model	Mask strategy	AP
SeqCo-DETR	$\text{Mask}_{online@30} + \text{Mask}_{\neg(online@30)}$	45.3
	$\text{Mask}_{online@50} + \text{Mask}_{\neg(online@50)}$	45.1
	$\text{Mask}_{online@70} + \text{Mask}_{\neg(online@70)}$	**45.8**
	$\text{Mask}_{online@75} + \text{Mask}_{\neg(online@75)}$	45.5
	$\text{Mask}_{online@80} + \text{Mask}_{\neg(online@80)}$	45.2

Table 5: Comparison of different proportions of the complementary mask added to both branches, evaluated on MS COCO `val2017`.

default result is 45.4, as reported in the paper. When there are different proportions of masks added to DETReg, the performance drops, indicating that the mask strategy is not suitable for DETReg. This is because DETReg is fully dependent on the hand-craft pseudo labels, which will be easily influenced. On the other hand, our SeqCo-DETR approach, which employs self-supervised learning to extract features, is suitable for the mask strategy. The combination of the image mask and the sequence consistency strategies can extract more representative contextual information about the object.

3.3.2 Different mask strategies for the two branches

Two branches with complementary masks We add complementary masks to both branches, as in the main paper. The complementary mask means the mask for each branch is strictly binary reversed. Thus, each branch will have a non-overlapped image view.

We conduct experiments with different mask proportions for each branch, and the results are presented in Tab. 5. For instance, $\text{Mask}_{online@30} + \text{Mask}_{\neg(online@30)}$ stands for the mask proportion for the online branch is 30% while for the momentum branch is 70%, where the mask for the momentum branch is the complementary mask of the online branch. From the table, the best proportion for the online branch is 70%, correspondingly, for the momentum branch is 30%, which is the complementary mask of the online branch.

Two branches with random masks We also explore the use of independent random masks for both branches, with each branch having the same or different proportions of masks. Especially, the mask for each branch is independently sampled.

As listed in Tab. 6, when the masks are independently sampled for both branches with different proportions, the best parameter is $\text{Mask}_{online@70} + \text{Mask}_{momentum@30}$, the corresponding result is 45.6, which is lower than the complementary mask, as 45.8. This proves the complementary mask strategy is better. We also try to add the same proportion masks for the two branches, but the results are inferior to those with different proportions or the complementary proportions of masks. These experiments prove that the complementary mask strategy is more suitable for our method.

Model	Mask strategy	AP
	$\text{Mask}_{online}@30 + \text{Mask}_{momentum}@30$	45.2
	$\text{Mask}_{online}@50 + \text{Mask}_{momentum}@50$	45.4
	$\text{Mask}_{online}@70 + \text{Mask}_{momentum}@70$	45.4
SeqCo-DETR	$\text{Mask}_{online}@20 + \text{Mask}_{momentum}@80$	45.1
	$\text{Mask}_{online}@30 + \text{Mask}_{momentum}@70$	45.1
	$\text{Mask}_{online}@70 + \text{Mask}_{momentum}@30$	**45.6**
	$\text{Mask}_{online}@75 + \text{Mask}_{momentum}@25$	45.5
	$\text{Mask}_{online}@80 + \text{Mask}_{momentum}@20$	**45.6**

Table 6: Comparison of different proportions of random mask added to both branches, evaluated on MS COCO `val2017`.

(a) 4 (b) 8 (c) 16 (d) 32 (e) 64 (f) 128

Figure 2: Illustration of different patch sizes of masked images.

3.3.3 The patch size of the mask

In this section, we investigate the patch size of the mask. The masked images are shown in Fig. 2. The image size is randomly resized between 320 and 480 during pre-training. The proportion of the image mask is fixed at 50% in the following experiments. As listed in Tab. 7, when the patch sizes are 8 or 16, the final results are the best. The final result seems less affected by the patch size, since the variance of the final result is small when patch sizes vary in a wide range. To choose the best value between patch sizes 8 and 16, we conduct other experiments. When the patch size is 8 and with the mask proportions of `Rnd(30,80)`, the result is 45.5, which is lower than 45.7 with patch size 16. When the patch size is 8 and with $\text{Mask}_{online}@70 + \text{Mask}_{\neg(online}@70)$, the result is 45.5, which is lower than 45.8 with 16. Based on the above experiments, we choose the patch size of 16 as the optimum parameter.

3.3.4 The feature mask

We try to add the mask to the features that are involved in calculating the self-supervised loss. Specifically, the feature mask is added to the output sequences from the projection head, and the feature mask is implemented by Dropout [□]. As listed in Tab. 8, the best

Patch size	4	8	16	32	64	128	Rnd(8,32)	Rnd(4,64)
SeqCo-DETR	45.3	**45.6**	**45.6**	45.4	45.4	45.4	45.3	45.5

Table 7: Comparison of different patch sizes of mask added to the online branch, evaluated on MS COCO `val2017`.

Feature mask proportion(%)	10	20	30	50	70
SeqCo-DETR	45.5	**45.6**	45.2	45.3	45.3

Table 8: Comparison of different proportions of feature mask added to the online branch, evaluated on MS COCO `val2017`.

Model	Mask strategy	AP
SeqCo-DETR	$\text{Mask}_{online@50} + \text{Mask}_{feature@20}$	45.2
	$\text{Mask}_{online@Rnd(30,80)} + \text{Mask}_{feature@20}$	45.2
	$\text{Mask}_{complementary@70-30} + \text{Mask}_{feature@50}$	45.1
	$\text{Mask}_{complementary@70-30} + \text{Mask}_{feature@20}$	45.2
	$\text{Mask}_{complementary@70-30} + \text{Mask}_{feature@10}$	45.3
	$\text{Mask}_{complementary@70-30} + \text{Mask}_{feature@1}$	45.4
	$\text{Mask}_{complementary@70-30} + \text{Mask}_{feature@0}$	**45.8**

Table 9: Comparison of different combinations of mask strategies, evaluated on MS COCO `val2017`.

proportion of feature mask value is 20%, with the result of 45.6.

Furthermore, we try to combine the feature mask and the image mask. We combine each best value of the image mask and the feature mask. As listed in Tab. 9, where $\text{Mask}_{online@50} + \text{Mask}_{feature@20}$ stands for the image mask with 50% proportion and feature mask with 20% proportion, $\text{Mask}_{complementary@70-30} + \text{Mask}_{feature@1}$ stands for the complementary mask with 70% proportion for online branch and the corresponding 30% proportion for the momentum branch and feature mask with 1% proportion. However, the results of the combination show a considerable performance drop compared with using each strategy alone. For example, when using $\text{Mask}_{complementary@70-30}$, i.e., $\text{Mask}_{complementary@70-30} + \text{Mask}_{feature@0}$, and $\text{Mask}_{feature@20}$ alone, the result is 45.8 and 45.6, respectively, but the combination result is only 45.2. Moreover, the results show that the final results become higher when there are fewer proportions of the feature mask. So it is not suitable to combine the image mask and the feature mask. One possible reason is that the combination would increase the difficulty of feature representation learning, which leads to performance drops. Thus, we only adopt the image mask strategy in the final version.

4 Visualization

As is shown in Fig. 3, we visualize the matched proposal pairs between the two branches. Only part of the proposals are visualized for clarity and brevity, and the matching indexes are derived from the bipartite matching. From the figure, the matching result is reliable between the two branches. The predicted bounding boxes from the two branches are on the same object, since the matching is mainly based on the location of the predicted bounding box. The mask added to the image has less affection for the matching, whereas the boundary of the predicted bounding box is more likely to be in the unmasked area, as shown in the right part of each image pair. Notably, we are not using the feature just inside a bounding box, but the feature from the sequence. The predicted bounding box is projected from the sequence, since each sequence stands for an object prediction. And the sequence has not only the feature of the object but also the global context information thanks to the characteristic of the

transformer. The visualization confirms the proposed method's ability to impose constraints on output sequences that predict the same object.

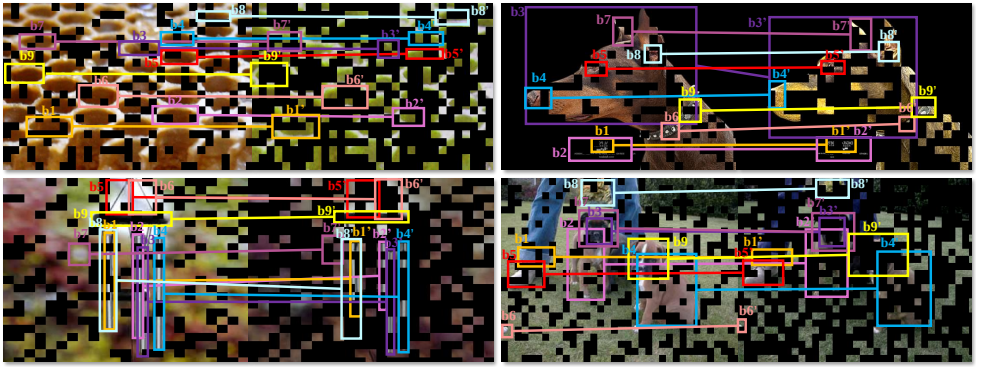

Figure 3: Visualization of paired proposals from different branches. The left is from the online branch, and the right is from the momentum branch.

5 Limitation and discussion

Our proposed method for sequence consistency training relies on bipartite matching, which requires the two views to share the same crop window location. Therefore, our method is limited when it comes to complex image augmentations. Additionally, while our method improves detection pre-training, it still requires the assistance of Selective Search and is not fully self-supervised. We believe that future research will overcome these limitations.

References

[1] Amir Bar, Xin Wang, Vadim Kantorov, Colorado J Reed, Roei Herzig, Gal Chechik, Anna Rohrbach, Trevor Darrell, and Amir Globerson. Detreg: Unsupervised pretraining with region priors for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14605–14615, 2022.

[2] Gary Bradski. The opencv library. *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, pages 120–123, 2000.

[3] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020.

[4] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9640–9649, 2021.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference*

on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA, pages 248–255, 2009.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[7] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.*, pages 303–338, 2010.

[8] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[9] 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.

[10] 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 *ECCV*, 2014.

[11] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[12] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104 (2):154–171, 2013.

[13] Xin Wang, Thomas Huang, Joseph Gonzalez, Trevor Darrell, and Fisher Yu. Frustratingly simple few-shot object detection. In *International Conference on Machine Learning*, pages 9919–9928. PMLR, 2020.

[14] Yizhou Wang, Meilin Chen, Shixiang Tang, Feng Zhu, Haiyang Yang, Lei Bai, Rui Zhao, Yunfeng Yan, Donglian Qi, and Wanli Ouyang. Unsupervised object detection pretraining with joint object priors generation and detector learning. *Advances in Neural Information Processing Systems*, 35:12435–12448, 2022.