Self-Supervised Pre-training with Transformers for Object Detection

Guoqiang Jin[1], **Fan Yang**[1], **Mingshan Sun**[1], **Yakun Liu**[1],
Wei Li[1], **Tianpeng Bao**[1], **Rui Zhao**[1,2], **Liwei Wu**[1]
[1]SenseTime Research
[2]Qing Yuan Research Institute, Shanghai Jiao Tong University, Shanghai, China
`{jinguoqiang, yangfan1, sunmingshan, liuyakun1, liwei1, baotianpeng, zhaorui, wuliwei}@sensetime.com`

Abstract

Self-supervised pre-training and transformer-based networks have significantly improved the performance of object detection. However, most of the current self-supervised object detection methods are built on convolutional-based architectures. We believe that the transformers' sequence characteristics should be considered when designing a transformer-based self-supervised method for the object detection task. To this end, we propose a novel transformer-based self-supervised learning method for object detection, which is based on the sequence consistency between the online and momentum branches. The proposed method minimizes the discrepancy of the output sequences of transformers with different image views as input and leverages bipartite matching to find the most relevant sequence pairs to improve the sequence-level self-supervised representation learning performance. Our method achieves state-of-the-art results on MS COCO (45.6 AP) and PASCAL VOC (63.9 AP), demonstrating the effectiveness of our approach.

1 Introduction

Object detection is a prediction-intensive process compared with the image classification task, which needs to locate and classify multiple objects in an image [1]. Existing deep learning-based object detection frameworks can be divided into one-stage methods [2, 3] and two-stage methods [4, 5], either of them requires hand-crafted components. Recently, the transformer-based detection method shows a new object detection paradigm [6, 7], which is a fully end-to-end method without hand-crafted components. Compared with convolutional-based architectures, transformer-based architectures define the problem as a sequence-to-sequence process; that is, the transformer converts the input to a sequence and processes the information in the form of a sequence, and the final output is also a sequence [8]. The transformer-based architectures do not rely on the inductive bias characteristics of convolutional-based architectures, such as locality and translation invariance, but rely on the global information processing procedure based on attention mechanism [8, 9]. However, the abovementioned detection methods require supervised training, which needs massive labeled data with extensive human labor since the labeling cost of object detection tasks is much higher than the image classification tasks.

Self-supervised representation learning could leverage unlabeled data efficiently [10, 11, 12, 13, 14]. By introducing customized pretext tasks, self-supervised representation learning could pre-train a model with unlabeled data. After that, the pre-trained model can be transferred or fine-tuned to solve specific tasks. Most self-supervised methods are designed for image classification tasks, which consider the image as a whole and only use image-level features [15]. However, object detection

36th Conference on Neural Information Processing Systems (NeurIPS 2022). Workshop: Self-Supervised Learning - Theory and Practice.

is a prediction-intensive task that requires predicting the location and category of multiple objects in one image, which needs object-level features. Therefore, directly applying these image-level self-supervised methods to objection detection tasks would lead to limited improvement. Some recent approaches [15, 16, 17, 18, 19, 20, 21, 22] utilize the inductive bias characteristics of the convolutional neural network(CNN) to achieve object-level self-supervised learning, which is unsuitable for the transformers. More recently, some transformer-based pre-training methods are starting to develop. UP-DETR [23] utilized random patches as the input to predict the location of the patches. Since the location of the patch is known, the proposed pretext task is more like a supervised method with pseudo labels. DETReg [24] relies on mimicking pre-trained model features; thus, the feature representation's ability is limited by the selected pre-trained model, limiting the model's performance.

To solve the aforementioned problems, we take advantage of the sequence characteristics of transformers and propose a self-supervised object detection pre-training method by maintaining the consistency of sequences from different views of an image. Each output sequence of the transformer decoder stands for an object prediction, which contains the location and category information of the object. Thus, the proposed pretext task addresses self-supervised learning on both the location and category of objects. Considering that the object prediction of each sequence varies under different image views, we propose to utilize bipartite matching [25] to get the optimal sequence pair to improve the sequence consistency learning process.

2 Method

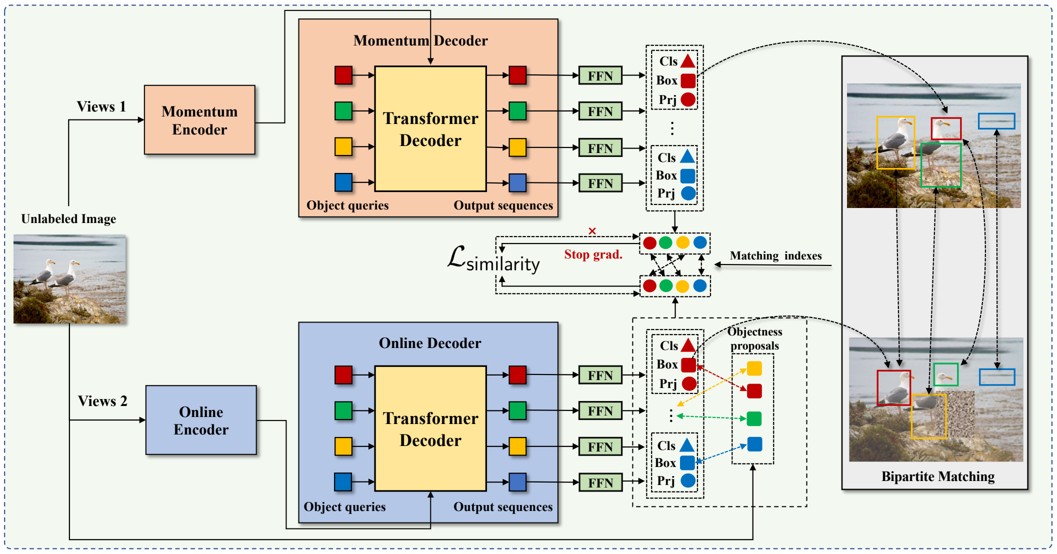

Figure 1: The proposed method contains two branches: the online and momentum branches. The Online/Momentum Encoder in the figure consists of a CNN backbone and a transformer encoder. The input of each branch is a different view of the image. Specifically, there is no location coordinates difference between the two views but only with content distortion difference. After being processed by the transformers, all object queries become the output sequences. Then the sequences pass three feedforward networks (FFN) to be predicted as the object's class, bounding box, and feature projection. There are two bipartite matching processes: one is between the outputs of two branches, and the other is between the output of online branches and the "objectness" proposals provided by the Selective Search. Finally, the similarity loss of each paired sequence feature projection is minimized to achieve sequence-level self-supervised representation learning.

The proposed method is designed to use the sequence characteristics to achieve self-supervised pre-training for object detection based on the transformers. The main idea of our method is to maintain consistency between sequences from differently augmented views of the same image. The framework is based on the Deformable DETR [7]. As shown in Fig. 1, we utilize the momentum design [10, 26, 27] to achieve the self-supervised learning. These two branches share the same structure, including the CNN backbone, transformer encoder, transformer decoder, and feedforward network (FFN) heads. Specifically, we add a projection head [12] after the transformer decoder to generate the feature projection for each sequence. After pre-training, the projection head will be removed, and the classification head will be reset and modified according to the downstream detection tasks. The rest of the weights learned during pre-training will be loaded as the initial weights during fine-turning. Following the momentum design, we add a stop gradient operator that the gradient only updates the weight of the online branch. And the weight of the momentum branch is updated by the momentum of the weight of the online branch, according to the formula:

$$\Theta_m \leftarrow \beta * \Theta_m + (1-\beta) * \Theta_o, \tag{1}$$

where Θ_m and Θ_o represent the parameters of the momentum branch and the online branch, respectively, β represents the updated momentum parameter.

The proposed sequence consistency strategy is simple and straightforward. Since each output sequence stands for an object prediction and each sequence contains the most relevant feature description for each object, we apply the consistency constraint on the sequences that predict the same object. Thus, we could maintain the object-level feature consistency instead of the image-level feature, which is more suitable for object detection tasks. The sequence consistency strategy is formulated as follows:

$$\mathcal{L}_{\text{ssl}} = \sum_{i=1}^{N} (\mathcal{L}_{\text{similarity}}(\mathbf{f}_{\text{ffn}}(\mathbf{s}_i), \widehat{\mathbf{f}}_{\text{ffn}}(\widehat{\mathbf{s}}_{\widehat{\sigma}(i)}))), \tag{2}$$

where $\mathbf{s}$ and $\widehat{\mathbf{s}}$ represent the sequences output by the transformer decoder from the momentum and online branches, respectively; $\mathbf{f}_{\text{ffn}}$ and $\widehat{\mathbf{f}}_{\text{ffn}}$ represent the FFN heads from the momentum and online branches, respectively; N is the number of sequences in one view, and $\widehat{\sigma}$ represents the matching relationship between the two sequences. $\mathcal{L}_{\text{ssl}}$ is the total self-supervised learning loss. $\mathcal{L}_{\text{similarity}}$ is a function that measures the similarity between sequences. We adopt $\mathcal{L}_2$ as the $\mathcal{L}_{\text{similarity}}$ loss.

As mentioned above, the image views for the two branches are different, so the output sequences from the branches would have diversity too. Thus, we adopt bipartite graph matching [25] to match the sequences that have the same object prediction from the online and momentum branches, which is defined as:

$$\mathcal{L}_{\text{match}}(\mathbf{y}, \widehat{\mathbf{y}}_{\sigma}) = \sum_{i=1}^{N} [-\lambda_{cm} \log \widehat{\mathbf{p}}_{\sigma(i)}(c_i) + \mathbb{1}_{\{c_i \neq \varnothing\}} (\lambda_{bm} \mathcal{L}_{\text{box}}(\mathbf{b}_i, \widehat{\mathbf{b}}_{\sigma(i)}))], \tag{3}$$

$$\widehat{\sigma} = \underset{\sigma \epsilon \sum_N}{\arg\min} \sum_{i}^{N} \mathcal{L}_{\text{match}}(\mathbf{y}_i, \widehat{\mathbf{y}}_{\sigma(i)}), \tag{4}$$

where N is the number of sequences in one view; $\mathcal{L}_{\text{match}}$ denotes the Hungarian matching loss; $\mathbf{y}$ and $\widehat{\mathbf{y}}$ are the predicted sequences from momentum and online branches, respectively; $\mathbf{b}$ and c are the location prediction and category prediction, respectively; $\widehat{\mathbf{p}}_{\sigma(i)}(c_i)$ is the probability of class c_i; $\widehat{\sigma}$ denotes the final optimal assignment; $\varnothing$ denotes the empty set; λ_{cm} and λ_{bm} are the corresponding weights, which are 2.0 and 5.0, respectively. After the output sequences, which predict objects at the same location, have been matched by the bipartite graph matching, the corresponding features from the sequences are used to calculate the $\mathcal{L}_{\text{ssl}}$. Specifically, the two image views share the same location augmentation parameters, and there are only content differences between the two views, such as color jittering, random erasing, and blur, which reduces the difficulty of the output sequences matching.

In order to get more rich semantic features rather than some background features from an image to calculate the self-supervised loss, the transformer needs to predict proposals that contain foreground objects as many as possible. Thus, we utilize the Selective Search [28] to initial foreground proposals to train the neural network to have the ability to predict "objectness" proposals. The Selective Search is an unsupervised method that could only generate the position of foreground proposals, not the category. Thus, there will be only two categories, i.e., foreground and background. The parameters of the Selective Search are the same as in DETReg [24].

The total loss consists of the region proposals loss $\mathcal{L}_{RPS}$ and the self-supervised learning loss on sequences $\mathcal{L}_{SSL}$, denoted as:

$$\begin{aligned}\mathcal{L}_{\text{total}}(\mathbf{y}, \widehat{\mathbf{y}}) =& \mathcal{L}_{RPS} + \mathcal{L}_{SSL} \\ =& \sum_{i=1}^{N} [\lambda_f \mathcal{L}_{\text{focal}}(c_i, \widehat{\mathbf{p}}_{\widehat{\sigma}(i)}) + \mathbb{1}_{\{c_i \neq \varnothing\}} (\lambda_b \mathcal{L}_{\text{box}}(\mathbf{b}_i, \widehat{\mathbf{b}}_{\widehat{\sigma}(i)})] + \sum_{i=1}^{N} [\lambda_e \mathcal{L}_{\text{ssl}}(\mathbf{z}_i, \widehat{\mathbf{z}}_{\widehat{\sigma}(i)}))],\end{aligned} \tag{5}$$

where $\mathcal{L}_{\text{focal}}$ is the focal loss of classification, $\mathcal{L}_{\text{box}}$ is the location loss of box, $\mathcal{L}_{\text{ssl}}$ is the self-supervised learning loss on sequence, and $\mathbf{z}_i$ stands for the output of $\mathbf{f}_{\text{ffn}}$. The $\mathcal{L}_{RPS}$ consists of $\mathcal{L}_{\text{focal}}$ and $\mathcal{L}_{\text{box}}$, which is the same as in DETReg [24], supervised by the proposals from Selective Search. λ_f, λ_b and λ_e are the weights of those three losses, which are set to 2.0, 5.0, and 10.0, respectively.

3 Experiments

The training of our method includes pre-training and fine-tuning stages. The pre-training dataset is ImageNet100 [29], which only contains 100 classes. The split of classes is the same as in DETReg [24]. The fine-tuning datasets are MS COCO [30] and PASCAL VOC [31]. In particular, we fine-tune the model on COCO `train2017` and evaluate it on COCO `val2017`. As for VOC, we fine-tune the model on VOC `trainval07+12`, and then evaluate on VOC `test07`. The comparison approaches are DETReg [24], UP-DETR [23] based on Deformable DETR, Deformable DETR [7] with different pretrained weights, and the common baseline Faster R-CNN [4]. Our training settings are the same with DETReg [24], more implementation details are presented in the Appendix.

Table 1: Comparison results on object detection datasets. †: We run the method on our codebase.

	COCO `val2017`			VOC `test07`		
Model	AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
Faster R-CNN [32]	42.0	62.1	45.5	56.1	82.6	62.7
Deformable DETR (Supervised CNN) [7]	43.8	62.6	47.7	59.5	82.6	65.6
Deformable DETR (SwAV CNN)†	45.0	63.8	49.2	61.0	83.0	68.1
UP-DETR (Deformable DETR) †	45.2	**64.2**	49.4	61.8	83.4	69.6
DETReg w/o feature embedding †	45.1	63.6	49.2	63.0	83.5	70.2
DETReg [24]	45.4	-	-	63.5	83.3	70.3
Ours	**45.6**	63.9	**49.7**	**63.9**	**84.1**	**71.1**

The experiment results on COCO and VOC are provided in Table 1. The above part of the table is the widely referenced baselines in object detection tasks, which are listed here for convenience. The middle part is our baseline, and the bottom part is our approach. Since the training process of DETReg is supervised by the pre-trained SwAV model [13], the pre-training process can be regarded as learning to use the fixed features from SwAV. On the contrary, our method proposes to use self-supervised ways to learn the features, which could help the network learn more discriminative features during pre-training. When the feature embedding part is removed in the experiment, the final results are 45.1 on COCO and 63.0 on VOC, which could be regarded as a baseline. For the final result, our method surpasses DETReg by 0.2 points on COCO and 0.4 points on VOC, which achieves state-of-the-art results on both datasets, proving that our learning-based method has better performance than the method that is supervised by manually defined pseudo-labels. Compared to the methods that only the backbone part is pre-trained, such as Deformable DETR (Supervised CNN) and Deformable DETR (SwAV CNN), our method could pre-train the entire object detection framework, and surpasses Deformable DETR (SwAV CNN) by 0.6 points on COCO and 2.9 points on VOC. Thus, our approach is more suitable for the object detection task. More results of the ablation study and visualization are presented in Appendix.

4 Conclusion

This paper proposes a novel transformer-based self-supervised learning method for object detection. The proposed method takes advantage of the sequence characteristics of transformers to achieve self-supervised learning by maintaining the consistency of the sequence under different image views. The bipartite matching is leveraged to get the optimal sequence pair to improve the efficiency of the sequence-level self-supervision. Experiments on MS COCO and PASCAL VOC prove the effectiveness of our method.

Acknowledgement This work is sponsored by Hetao Shenzhen-Hong Kong Science and Technology Innovation Cooperation Zone (HZQB-KCZYZ-2021045).

References

[1] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, 2020.

[7] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. 2021.

[8] 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.

[9] 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.

[10] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[11] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. *arXiv preprint arXiv:2104.02057*, 2021.

[12] 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.

[13] 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.

[14] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

[15] Enze Xie, Jian Ding, Wenhai Wang, Xiaohang Zhan, Hang Xu, Zhenguo Li, and Ping Luo. Detco: Unsupervised contrastive learning for object detection. *CoRR*, 2021.

[16] Songtao Liu, Zeming Li, and Jian Sun. Self-emd: Self-supervised object detection without imagenet. *CoRR*, 2020.

[17] Ceyuan Yang, Zhirong Wu, Bolei Zhou, and Stephen Lin. Instance localization for self-supervised detection pretraining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3987–3996, 2021.

[18] Jiahao Xie, Xiaohang Zhan, Ziwei Liu, Yew-Soon Ong, and Chen Change Loy. Unsupervised object-level representation learning from scene images. *CoRR*, 2021.

[19] Byungseok Roh, Wuhyun Shin, Ildoo Kim, and Sungwoong Kim. Spatially consistent representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1144–1153, 2021.

[20] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3024–3033, 2021.

[21] Di Wu, Siyuan Li, Zelin Zang, Kai Wang, Lei Shang, Baigui Sun, Hao Li, and Stan Z Li. Align yourself: Self-supervised pre-training for fine-grained recognition via saliency alignment. *arXiv preprint arXiv:2106.15788*, 2021.

[22] Zhenda Xie, Yutong Lin, Zheng Zhang, Yue Cao, Stephen Lin, and Han Hu. Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16684–16693, 2021.

[23] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. UP-DETR: unsupervised pre-training for object detection with transformers. In *CVPR*, pages 1601–1610, 2021.

[24] 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. *arXiv preprint arXiv:2106.04550*, 2021.

[25] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[26] Mengde Xu, Zheng Zhang, Han Hu, Jianfeng Wang, Lijuan Wang, Fangyun Wei, Xiang Bai, and Zicheng Liu. End-to-end semi-supervised object detection with soft teacher. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3060–3069, 2021.

[27] Yen-Cheng Liu, Chih-Yao Ma, Zijian He, Chia-Wen Kuo, Kan Chen, Peizhao Zhang, Bichen Wu, Zsolt Kira, and Peter Vajda. Unbiased teacher for semi-supervised object detection. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] Jinhong Deng, Wen Li, Yuhua Chen, and Lixin Duan. Unbiased mean teacher for cross-domain object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4091–4101, 2021.

[35] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. *arXiv preprint arXiv:2203.16527*, 2022.

A Appendix

A.1 Overview

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

A.2 Implementation Details

Training details. In the pre-training stage, we add an additional FFN head for projecting sequence features. The FFN is composed of 2 hidden layers, and the number of hidden layer neurons is 256. In addition, since all the proposals generated by Selective Search are foreground boxes, the classification head only has 2 categories, i.e., foreground and background. The backbone of the proposed method is ResNet-50 [33], which is initialized by SwAV [13], which is the same as in DETReg [24]. Following DETReg, the number of epochs in pre-training 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. The parameters in the fine-tuning stage are also the same as in DETReg. On the COCO, the epoch 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 the VOC, the epoch 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. Experiments are carried out on 8 * NVIDIA V100 GPUs.

Image augmentations. In order to generate different image views for the two branches, following [34], we adopt weak image augmentations for the momentum branch and strong augmentations for the online branch. To ensure the two views have location consistency, the image $view\ \#2$ is partially built based on the $view\ \#1$. We first generate a base image view from the unlabeled input image, using random flips, random resize, and random resized-crop, as $view\ \#1$. The base image view is the same as in DETReg [24]. As for the $view\ \#2$, we add more augmentations to the base image view, including color jitter, random grayscale, and random blur. Thus, there is no location coordinates difference between the two views.

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

A.3 Ablation Study

Sequence utilization methods. As mentioned before, considering the importance of the sequences that are output by transformers, we design a sequence consistency strategy that maintains the output sequence consistency between the online and momentum branches to achieve the self-supervised pre-training. One of the most naive strategies is the one-by-one match, because the sequence output by transformers has sequential characteristics. However, different branches have different input image views and slightly different network parameters. Even if the output sequences are in the same order, their predicted results are still different. Therefore, the bipartite matching is adopted to match the sequences from different branches which predict the same object. As shown in Table. A1, bipartite matching could improve results from 45.1 to 45.6 compared to one-by-one matching, proving the bipartite matching method's effectiveness. Meanwhile, in order to achieve a more sufficient supervision of the sequence, we try to use the outputs of classification head f_{cls}, regression head f_{box} and projection head f_{prj} in Eq. (2) at the same time. It can be seen from the table that the fusion of multiple heads decreases the final results in both matching settings. It may be because the self-supervision on the projection head is enough; redundancy supervision on three heads would cause a performance drop.

Table A1: Comparison of sequence utilization strategies, evaluated on MS COCO `val2017`.

Model	One-by-one matching	Bipartite matching	Multi-feature	AP
Ours	✓			45.1
	✓		✓	45.2
		✓	✓	45.1
		✓		**45.6**

Algorithm 1 Pseudo code 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, more details are in Sec. A.2
# 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()
```

Table A2: Comparison of pre-training datasets and region proposal strategies, evaluated on MS COCO `val2017`. †: We run the method on our codebase.

Model	IN100	IN100(Rnd bbox)	COCO	COCO+	COCO GT
DETReg †	45.4	44.1	45.1	45.1	45.6
Ours	**45.6**	**44.1**	**45.5**	**45.6**	**45.7**

Pre-training datasets and region proposal strategy We conduct several experiments to compare the performance with different pre-train datasets. IN100 stands for the baseline ImageNet100, `Rnd bbox` stands for the random proposals. COCO stands for the COCO `train2017`, COCO+ denotes the COCO `train2017` plus the COCO `unlabeled` dataset. The Selective Search is used to generate initial region proposals for COCO and COCO+. COCO GT stands for the COCO `train2017`, where the region proposals come from the COCO ground truth. Specifically, COCO is a multi-object dataset, while ImageNet is a single-object dataset. As shown in Table A2, our method achieves better results against DETReg on both single-object and multi-object datasets, which proves that self-supervised learning based has better generality over different types of datasets. Furthermore, we compare the influence of different region proposal strategies. When using the less "objectness" proposals, i.e., random proposals, both method drops a lot. When using the ground truth as the region proposal, DETReg improves from 45.1 to 45.6, whereas ours improves from 45.5 to 45.7. This proves that the method that is fully dependent on the hand-craft pseudo labels would be easily affected by the quality of the pseudo labels, whereas the learning-based method has less affected by the pseudo labels. The quality of the region proposals generated by Selective Search is sufficient for the proposed method to learn useful information.

Table A3: Comparison of self-supervised loss strategies on MS COCO `val2017`.

Model	Self-supervised loss strategy	AP
Ours	L1 loss	45.3
	L2 loss	**45.6**
	MoCo loss	45.4
	Predictor + BYOL loss	45.3

Table A4: Comparison of loss coefficient strategies on MS COCO `val2017`.

Model	Loss coefficient strategy	AP
Ours	Without $\mathcal{L}_{RPS}$	44.6
	Turning off $\mathcal{L}_{RPS}$ at 20 epochs	45.3
	Turning off $\mathcal{L}_{RPS}$ at 40 epochs	45.4
	Exponential decaying $\mathcal{L}_{RPS}$	45.5
	Default using $\mathcal{L}_{RPS}$ for 50 epochs	**45.6**

Self-supervised loss strategy. As shown in Table A3, we compare the effects of different self-supervised losses. Predictor + BYOL loss follows the BYOL [14] 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 MoCov3 [11] symmetric version. Surprisingly, L1 and L2 loss can achieve satisfactory results. Thus, the L2 loss is adopted in the experiments.

The comparison of different loss coefficient strategies is shown in Table A4. We study the influence of $\mathcal{L}_{RPS}$ in Eq. (5). As noted before, $\mathcal{L}_{RPS}$ is the region proposal loss, which is supervised by the results from Selective Search. When there is no $\mathcal{L}_{RPS}$ during pre-training, the result is only 44.6 on COCO. As a contrast, the result of Deformable DETR (SwAV) is 45.0. Since the default prediction box is in the center of the image, the model's performance drops compared to SwAV which uses information from the whole image. When decaying or turning off $\mathcal{L}_{RPS}$ during pre-training, the final results also decrease. This means that $\mathcal{L}_{RPS}$ is important in the pre-training stage because the self-supervised training is dependent on rich semantic features from proposals, which have to contain foreground objects as many as possible. Stopping or decaying to learn foreground proposals during training will affect the self-supervised learning process of the network.

A.4 Visualization

As is shown in Fig. A1, we visualize the matched predicted proposal pairs between the two branches. Only part of the proposals are visualized for clarity and brevity. From the figure, the matching result is reliable between the two branches. The paired 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 visualization proves the proposed method could achieve correct constraints on output sequences that predict the same object.

A.5 Discussion

In the bipartite matching for the sequence consistency training, the two views need to share the same crop window location, which limits our method with complex image augmentations. We have not achieved a fully self-supervised detection pre-training, but we still rely on the Selective Search's help. We believe future works will break these limitations.

Recent work [35] has proven that an object detection method that is entirely constructed on the transformer without a CNN backbone is feasible. Our self-supervised pre-pretraining method is based on the sequence characteristics of transformers. Thus, it would be applicable to the fully transformer-based detection framework. We will explore the feasibility of this approach in the future.

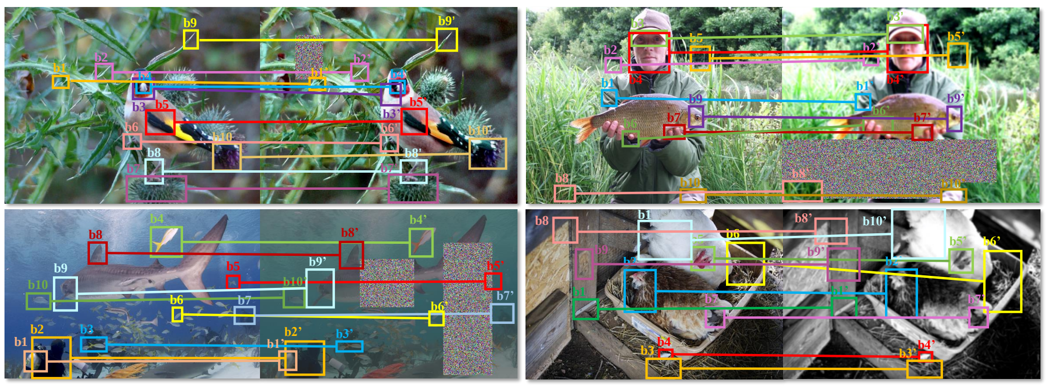

Figure A1: Visualization of paired proposals from different branches with bipartite matching. The left is from the momentum branch, and the right is from the online branch.