Supplementary Material for "CoCo-RCNN"

1 Analysis of End-to-End Object Detector

We choose Sparse-RCNN [10] as our baseline for FSOD and there are six heads. We first pre-train a model on MS COCO 14 train dataset where only the annotation of base classes are provided, and then conduct the analysis on MS COCO 14 val dataset with 60 classes.

For each proposal, the detector degrades at one head when the matching cost between the prediction and groundtruth annotation is larger than the cost at the previous head. In practice, when the prediction is degraded at the current proposal, we observed that the ground truth annotation will be assigned to the prediction of another proposal after bipartite matching. As such, for the convenience of implementation, for each ground truth annotation, we count one prediction degradation when the assigned proposal after bipartite matching changes. For each annotation, the maximum total number that a detector degrades is then 5 when there are in total 6 heads.

Meanwhile, for each annotation, purely counting the total times of degrading is not enough. For example, for the cases where a detector degrade once, *e.g.*, [a,b,b,b,b,b] and [a,a,a,b,b,b] where a,b denote the index of assigned proposal, the former means the detector performs better in concentrating on the detected objects. As such, we specifically assign the fine grained value for the following cases:

- For [a,b,b,b,b,b] and [b,b,b,b,b,a], we set the degrading times as 0.2.
- For [a,a,b,b,b,b] and [b,b,b,b,a,a], we set the degrading times as 0.6.
- For [a,a,a,b,b,b], we set the degrading times as 1.
- For [a,b,c,c,c,c] and all of its permutation variants (*e.g.*, [a,c,c,c,c,b], [c,c,c,c,a,b]), the degrading times is 1.2.
- For [a,a,b,b,b,c] and all of its permutation variants, we set it as 1.6.
- For [a,a,b,b,c,c], we set the degrading times as 2.
- For [a,b,c,d,d,d] and all of its permutation variants, we set it as 2.5.
- For [a,a,b,b,c,d] and all of its permutation variants, we set it as 3.
- For [a,a,b,c,d,e] and all of its permutation variants, we set it as 4.
- For [a,b,c,d,e,f] and all of its permutation variants, we set it as 5.

For Fig. 1(b), we first calculate the total times that the detector degrades for each groundtruth annotation and then average them among all annotations for each class. From Fig. 1(b), for the class which achieve high AP, the total times that the detector degrade is generally low.

Besides the analysis on MS COCO 14, we observed similar trend on the model trained on MS COCO 17 (provided by Sparse-RCNN github repo).

2 Approach Summary and Future Direction

2.1 Approach summary

CoCo regularization aims to facilitate the adaptation of end-to-end object detectors for new classes in few-shot finetuning. It first A) applies supervised contrastive learning among object encodings to make the detector can concentrate on the detected objects. By comparing CoCo-RCNN and alternative methods (Table 3 Row_{4-7}) with baseline (Table 4 Row_1), the performance is improved as the encodings are more discriminative. Secondly, since the proposals in end-to-end object detectors do not have spatial priors (*i.e.*, anchor box), to calculate the contrastive loss, CoCo regularization B) includes a specific strategy to properly select positive and negative encodings for each class, which serves two purposes: B1) *To avoid interfering the training of end-to-end detector*, we use the groundtruth assigned to predictions $\{f_d^h(\mathbf{o}_n^h)\}_{n=1}^{N_P}$ at each head as reference. The assignments at each head are calculated dynamically ($\text{Line}_{242-246}$) and are essential for training while manually setting consistent assignment hurts the training clearly ($\text{line}_{488-492}$, Table 3 $\text{Row}_{1,2}$). B2) *To avoid confusion cases where the assigned labels for one proposal at different heads are inconsistent*, at h-th head, we use assigned labels at $(h, h-1)$-th heads and Eq.(2-3) as a selection criterion, whose benefits is discussed in Table 3 Row_{4-7}. As illustrated in Fig.2(b), $\mathbf{o}_n^h$ is included in CoCo regularization while $\mathbf{o}_n^{h+1}$ is not since the assigned labels have confusion. Lastly, since the proposal vectors are class-irrelevant and the model is difficult to concentrate on objects at lower heads, we also embed class-relevant information by adding each of the class encodings on a sub-group of proposal vectors. Eq.(3) includes false-negative cases when only Eq.(2) is used ($\text{Line}_{564-575}$) and improves training efficiency. However, Eq. (3) is only necessary for large-scale training.

2.2 Adaptation to tiny objects

Applying CoCo regularization can achieve better detection precision for both small and large objects. However, detecting small objects is commonly challenging since small objects have *lower resolution resulting in less distinctive regions*. In addition, they can be deformable or overlapped by other objects. Then, it is hard for the correlation module f_a^h to generate discriminative object encodings. As such, *less* encodings of the small objects can be selected, and the performance gain by CoCo-RCNN in detecting large objects nAPl is more clear than the gain for nAPs.

Regarding small object detection (*e.g.*, LVIS), an object can be better detected when more background/contextual information is included [5]. DCNet [4] sets a distillation framework to make the model more aware of the context while TIP [7] designs a transformation invariant module to make the network robust to deformable instances, leading to higher nAPs. As such, we can adopt ideas in [2, 4, 7] to set regularization to improve the performance. We can also expand the bbox of the predicted proposal or use ViT [3] backbone to have a more

context-aware representation. Furthermore, we can set less constraint in selecting object encodings for small objects (*e.g.*, Eq.(3) with $l^h(n,t) \leq l^{h-1}(n,t) - \Delta$ where $\Delta > 0$) and include more object encodings for small objects in CoCo regularization.

3 Experimental Study Details

3.1 Datasets

We follow the protocol defined in the TFA [11]. For an K-shot object detection task, k annotations for each class are provided. From TFA, a random set of annotations for novel classes are pre-selected by specifying the random seed. Then, k annotations for each base classes are also selected. As such, the model is finetuned on KN_C annotations where $N_C = |\mathcal{C}_{novel} \cup \mathcal{C}_{base}|$.

The annotations for each class is selected by specifying the random seed. To avoid bias in data selection, under the same base-novel partition of one dataset, we repeat the experiments (runs) multiple times. The annotations used for finetuning provided in different runs vary from each other while the adapted models are then evaluated the on the same test set. Under each partition, we do 10 runs and report the averaged value.

3.2 Baseline model selection

As mentioned in the related work, there are three popular choices of the end-to-end object detector frameworks, Detr [1], Deformable Detr [12], and Sparse-RCNN [10]. For both Detr and Deformable Detr, they use cross-attention mechanism to obtain object encodings for each of the proposal encodings. Instead, the Sparse-RCNN uses dynamic instance interaction module, which first extracts a pooled feature from feature maps through RoI pooling and then applies dynamic convolution to calculate object encodings. Within the dynamic convolution, the value of convolution kernels depends on the proposal encodings. In contrast, within Faster-RCNN, the pooled feature is then processed by a fixed convolution blocks to obtain the object encodings where the FSCE [9] then minimizes the SupCT loss on the object encodings.

For an end-to-end object detector, the objective is to model the relationship between the proposal encoding and the features extracted from image feature maps. For our CoCo-RCNN, we specifically study how to make the detector concentrate on the previously detected objects. Then, we choose to use supervised contrastive learning and keep the object encodings concentrated by increasing the discriminability. Then, one future work is to provide more effective regularization by including other perspective. We also notice the FSCE [9] makes modification on SupCT [6] while they only aim to extract discriminative features. In this way, we do not utilize the pooling characters in Sparse RCNN but we hope our constantly concentrated encoding regularization can provide a general solution to the representation learning of end-to-end object detectors.

3.3 Regarding Class Encoding

As mentioned in Section 3.1, the finetuning set within each run is sampled from the union of base and novel classes. As such, in CoCo-RCNN, given the post-processed class encodings $\{\mathbf{s}_c\}_{c\in\mathcal{C}_{novel}\cup\mathcal{C}_{base}}$, to detect objects for each image, we first randomly split all proposal vectors into $|\mathcal{C}_{novel}\cup\mathcal{C}_{base}|$ groups without overlapping. Then, for each group, a class encoding is added to each of the grouped proposal vectors.

As indicated in Table 5, though the split of proposal vectors are random, the variance induced in the final detection performance is quite small. As such, during evaluation, for each run, we randomly split the proposal vectors before detecting one image and we only perform detection for each image once.

3.4 Ablation study on Constantly Concentrated Encoding

In Table 3, we summarize the performance between different regularization methods. The *hard-deepest* and *hard-lowest* force label-prediction assignment across all heads to be consistent, they in effect directly change the way that the end-to-end object detector used to calculate detection losses at each heads. However, the detection losses are kept in the other alternatives.
1) Compared with the baseline (Row_1 in Table 4), by encouraging the model to focus on the previously detected regions, the performance can be generally improved. However, the performance gain from *distillation* is the smallest.
2) Meanwhile, for the fair comparison between the alternative *contrastive* and our full method, we still use the same sampling strategy of object encodings at the first head as our full method.

3.5 Ablation study on the full method

In Table 4, we summarize the ablation study of our full method and use "2nd-6th heads", "Class Encoding" and "Negative' as three components.

"2nd-6th heads" specifically indicates the supervised contrastive learning across heads. When the "Class Encoding" is not added, we can only sample object encodings from the 2nd-6th heads to calculate SupCT loss. When the "Class Encoding" is checked, we sample the object encodings from all heads.

"Class Encoding" indicates adding class encodings to the proposal vectors. When the class encoding is checked while "2nd-6th heads" is not, it means we only add the "Class Encoding" to the proposal vectors and we still finetune the detector using the vanilla detection losses.

"Negative" means sampling object encodings for the SupCT loss calculation. From the table, we can see that including the negative object encodings can improve the detection precision and mitigate the inefficient training due to the limitation of pairs during SupCT loss calculation.

4 Comparison with other methods

In this paper, we mainly study the regularization needed in end-to-end object detector training (large-scale training and few-shot finetuning). With the constantly concentrated encoding regularization, we can achieve clear performance gain w.r.t. the finetuning baseline. We also compare our method with a few popular finetuning-based adaptation method and perform fair comparison.

Besides, we notice the recent work DefRCN [8] has achieved significant performance improvement on few-shot object detection. DefRCN differs from our method as it uses a different feature backbone as well as a different detection framework.

DefRCN employs the two-stage object detection framework, *e.g.*, Faster-RCNN, and uses ResNet-101 (without FPN) as backbone. When the FPN is not added, the detector generate a pooled feature from 'C4' (the feature maps after the penultimate convolution blocks) and then feeds the pooled feature into the last convolution block to generate an object encoding (denoted as 'C4'). When the FPN is added, the detector obtains the pooled features from the features after FPN (denoted as 'FPN'). The pooled feature, serving as an object encoding, will then be fed into the detection module. Then, before pre-training the object detector, we first initialize the ResNet-101 using imagenet-pretrained model. However, the FPN is still trained from scratch.

As an FPN is trained from scratch, the FPN can easily overfit to the detection of base objects. The pooled feature can be used to detect base objects perfectly but may fail in detect novel instances, which will hurt the adaptation performance. In contrast, when FPN is excluded, all of the parameters used to extract object encodings are updated from the values in the imagenet-pretrained models. As the model has been pretrained to classify the instances of novel classes, the pooled feature is discriminative for novel classes in nature and the pooled feature can be less overfit to the base object instances.

Table S1. K-shot classification accuracy on MS COCO.

Method	1-shot		5-shot		10-shot	
	base	novel	base	novel	base	novel
ImageNet-pretrained	32.8	44.7	47.5	63.1	52.3	65.4
FPN	72.5	35.0	80.5	52.3	82.8	54.8
C4	50.8	32.8	63.9	50.0	66.2	53.4

To demonstrate it, we study the extracted object encodings to analyze the backbones. For each backbone, we first use the groundtruth annotations to extract the object encodings for each instance. To note, the instances are from the union of finetuning set and evaluation set. For the objects in finetuning set, we average their encodings for each class as the class prototype. For the object encodings from evaluation set, we then calculate the classification accuracy.

We follow the protocol of few-shot classification and perform prototype classification directly, *i.e.*, for each encoding of the evaluation set, we calculate its cosine similarity with each of the class prototypes where the class with highest similarity is assigned to the sample. As the test set show imbalanced distribution, *e.g.*, too many samples for class 'people' and only a few instances for 'sandwich', we first calculate the classification accuracy for each class and then obtain the averaged accuracy among classes.

We use MSCOCO as evaluation dataset to better show the effect of overfit. As shown in Table S1, we classify each encoding in the evaluation set among 80 classes and then average the classification accuracy for base class set and novel class set separately. The backbone model we used are the Resnet-101 pretrained on imagenet (denoted as ImageNet), the Resnet-101 with FPN after pre-training on base classes (denoted as FPN), the Resnet-101 after pre-training on base classes (denoted as C4). For "ImageNet-pretrained", we only use the ResNet-101 w/o FPN and use the same way to extract object encodings as C4.

As shown in Table S1, after training, the classification accuracy on novel classes drop. However, the FPN clearly overfits to the base classes: the base classification accuracy by FPN is much higher than that of C4 while the novel classification accuracy for C4 and FPN are similar. Since the FPN has been overfitted to the base classes, the efficiency of few-shot adaptation on novel classes is hurt. As such, we fail to see clear performance gain when the Gradient Decoupled Layer (GDL) is applied to FPN during adaptation.

Thus, even though we value the idea behind GDL, we still think the performance gain of GDL is closely related to the case when the FPN layer is excluded. Then, all parameters involved in extracting object encodings are tuned from a pre-trained model and is not severely overfit to the base objects. In contrast, even though our method underperforms DefRCN, we value our method as it specifically targets on the adaptation of end-to-end object detectors and study the regularization needed in finetuning. In this way, CoCo-RCNN help learn a better representation of object encodings and it is even useful under large-scale training.

References

1. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European conference on computer vision. pp. 213–229. Springer (2020)
2. Dai, Z., Cai, B., Lin, Y., Chen, J.: Up-detr: Unsupervised pre-training for object detection with transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1601–1610 (2021)
3. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2020)
4. Hu, H., Bai, S., Li, A., Cui, J., Wang, L.: Dense relation distillation with context-aware aggregation for few-shot object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10185–10194 (2021)

5. Hu, P., Ramanan, D.: Finding tiny faces. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 951–959 (2017)
6. Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., Krishnan, D.: Supervised contrastive learning. Advances in Neural Information Processing Systems **33**, 18661–18673 (2020)
7. Li, A., Li, Z.: Transformation invariant few-shot object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3094–3102 (2021)
8. Qiao, L., Zhao, Y., Li, Z., Qiu, X., Wu, J., Zhang, C.: Defrcn: Decoupled faster r-cnn for few-shot object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8681–8690 (2021)
9. Sun, B., Li, B., Cai, S., Yuan, Y., Zhang, C.: Fsce: Few-shot object detection via contrastive proposal encoding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7352–7362 (2021)
10. Sun, P., Zhang, R., Jiang, Y., Kong, T., Xu, C., Zhan, W., Tomizuka, M., Li, L., Yuan, Z., Wang, C., et al.: Sparse r-cnn: End-to-end object detection with learnable proposals. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14454–14463 (2021)
11. Wang, X., Huang, T., Gonzalez, J., Darrell, T., Yu, F.: Frustratingly simple few-shot object detection. In: International Conference on Machine Learning. pp. 9919–9928. PMLR (2020)
12. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. In: International Conference on Learning Representations (2020)