[Supplementary] NIFF: Alleviating Forgetting in Generalized Few-Shot Object Detection via Neural Instance Feature Forging

A. Implementation Details

A.1. Generator Training

Using the base trained RoI head parameters and the gathered statistics in the data watchers, we train the generator for 2k iterations. We optimize the generator using SGD with a batch size of 600 features, momentum of 0.9, and weight decay of $5e^{-5}$. The learning rate is set to 0.001. The scaling factor for the KL divergence loss is set to $\lambda_{\text{KL}} = 5$.

A.2. Novel Training

During novel training, the model is also optimized using SGD with batch size of 16 and learning rate of 0.005 and 0.01 for MS-COCO and PASCAL-VOC, respectively. We set the warmup iterations to 200. Step decays are performed at 2500, 4000, and 6400 for MS-COCO 5, 10,and 30-shot settings, respectively. As for PASCAL-VOC, we perform the decay for the first three shot settings at 1000 and 1500 for the 5 and 10-shot settings. To perform the distillation, we unfreeze the RoI head and down scale its learning rate by a factor of 0.015. We set the scaling factors as follows: $\lambda_{\text{F}} = 0.1$ and 0.01 for the mEWC penalty term.

B. Further Ablations

Data Watcher Config.	10-Shot Inference					
	AP	bAP	nAP	AR	bAR	nAR
(1) After Act.	32.0	36.7	18.0	29.9	32.7	21.5
(2) Before FBN	**32.2**	**36.9**	18.1	**30.1**	**33.0**	21.1
(3) Both	**32.2**	36.6	**19.1**	29.6	32.1	**22.3**

Table 1. Where to place the data watchers to record useful statistics for better feature generation with respect to the frozen BatchNorm layers and the activations that follow. Results on MS-COCO 10-shot.

Which statistics matter? In Tab. 1, we study which statistics are needed to capture the base data distribution. To this end, we place data watchers in different places and train different generators accordingly. Then we finetune on the novel data in each setup and report the final detection results. The different locations for placing data watchers are: (1) after the activations (Act.) following the frozen-BN (FBN), (2) before the FBN layers, (3) both locations. As we can observe, locations (2) and (3) yield empirically better results than (1). Although the AP of (2) and (3) is the same, we choose (3) as its nAP is slightly higher.

	Model Configuration	10-Shot Inference					
		AP	bAP	nAP	AR	bAR	nAR
0	DeFRCN	30.6	34.6	18.6	29.1	32.0	20.5
A	DeFRCN (Base-Free)	18.2	18.5	17.4	17.5	16.2	21.3
B	+ Generator	30.7	35.0	17.8	28.6	31.3	20.9
C	+ CFA	32.0	36.4	18.5	29.6	32.4	21.3
D	+ DA	**32.2**	**36.8**	18.4	**29.9**	**32.6**	21.7
E	+ mEWC (NIFF)	**32.2**	36.6	**19.1**	29.6	32.1	**22.3**

Table 2. Incremental component analysis on MS-COCO (10-shot).

Component analysis. In Tab. 2, we start by reporting the results of the state-of-the-art model in transfer-learning namely, DeFRCN [5] (Config **0**)). In Config **A**, we train a DeFRCN model on novel data without using the base classes. We consider this as our baseline, to which we gradually introduce our contributions in an incremental fashion. Note that without using base data, DeFRCN performance drops dramatically on the base classes (by 40.5%). In Config **B**, we train our standalone lightweight generator and finetune DeFRCN on novel data while replaying synthetic base features from the generator. We show that we are able to almost recover the overall performance of DeFRCN in Config **0**, all without using any base data. Next, in Config **C**, we apply CFA [2] on the gradients which are backpropagated on the RoI head. In Config **D** and **E**, we add the chosen pixel-level data augmentation (DA) and parameter-level (mEWC) regularization techniques, which allow us to achieve state-of-the-art results in the overall performance.

B.1. Generator Architecture

In Tab. 3, we investigate the impact various architectural design choices on the overall detection performance. The utilized generator throughout this work comprises five convolutional layers ($L = 5$) with kernel size ($K = 3$) and an input noise dimension ($z = 100$). To study the effect of each of these design choices, we change only one factor and leave the rest unchanged. Firstly, increasing the number of layers contribute to higher overall performance where the base performance is noticeably improved. However, past

Model Configuration	10-Shot Inference			
	AP	bAP	nAP	AR
Number of Layers (L=3)	27.6	31.3	16.4	27.6
Number of Layers (L=5)	30.7	35.0	17.7	28.8
Number of Layers (L=7)	31.2	35.7	17.7	29.3
Number of Layers (L=10)	31.1	35.6	17.5	29.2
Kernel Size (K=1)	30.2	34.7	17.7	28.3
Kernel Size (K=3)	30.7	35.0	17.7	28.8
Kernel Size (K=5)	30.5	34.8	17.6	28.7
Kernel Size (K=7)	30.6	34.8	17.8	28.6
Noise Dimension (z=50)	30.5	34.9	17.6	28.5
Noise Dimension (z=100)	30.7	35.0	17.7	28.8
Noise Dimension (z=1000)	30.7	35.0	17.7	28.8

Table 3. Impact of various generator architectural design choices on MS-COCO (10-shot). We finetune DeFRCN without base data using the generator without any regularization techniques.

$L = 7$ the performance starts to slightly drop. Secondly, we show that increasing the kernel size do not result in any performance gain. Finally, increasing the noise dimension past $z = 100$ yields no change in the performance implying that the generator is able to generate diverse high-quality features without requiring high-dimensional noise vectors.

B.2. mEWC VS EWC

Configuration	10-Shot Inference			
	AP	bAP	nAP	AR
EWC ($\lambda_{\text{EWC}} = 1.0$)	33.0	38.1	17.6	30.3
EWC ($\lambda_{\text{EWC}} = 0.1$)	32.9	38.2	17.1	30.6
EWC ($\lambda_{\text{EWC}} = 0.01$)	32.2	37.7	15.9	29.8
EWC ($\lambda_{\text{EWC}} = 0.001$)	31.8	36.7	17.3	29.7
mEWC ($\lambda_{\text{EWC}} = 1.0$)	33.0	38.5	16.5	30.5
mEWC ($\lambda_{\text{EWC}} = 0.1$)	33.0	38.4	16.6	30.5
mEWC ($\lambda_{\text{EWC}} = 0.01$)	32.2	36.6	19.1	29.8
mEWC ($\lambda_{\text{EWC}} = 0.001$)	30.3	33.9	19.3	28.8

Table 4. A comparison between the EWC with the full Fisher matrix VS the utilized mEWC with mean Fisher matrix on NIFF-DEFRCN on MS-COCO (10-shot). Further, a study on the impact of scaling EWC/mEWC regularization term.

In Tab. 4, we show the difference between the vanilla EWC [4] with the full Fisher matrix and the utilized mean EWC (mEWC) using a mean Fisher matrix per parameter. Moreover, we show the effect of various scaling factors λ_{EWC} when applying the EWC/mEWC penalty term during novel training. The lower the scaling factor, the more changes in parameters are allowed. Firstly, we observe that EWC can better maintain the base performance along different scaling factors at the price of the novel performance. Secondly, as we reduce the scaling factor for mEWC, the base performance drops compared to EWC. However, we achieve the same AP as EWC at $\lambda_{\text{EWC}} = 0.01$ with a lower bAP and a higher nAP. Hence, we use this setting across our experiments. The user can decide on the bAP and nAP trade-off according to the application. Finally, we observe that in mEWC, the nAP improves with lower λ_{EWC}, however, at the cost of lower bAP.

B.3. Novel Training Loss Components

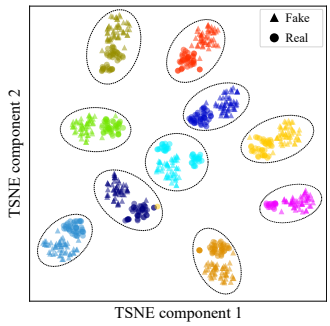
Model Configuration	10-Shot Inference			
	AP	bAP	nAP	AR
DeFRCN w/G.	30.7	35.0	17.7	28.8
w/o $\mathcal{L}_{\text{conf}}$	28.9	32.6	17.7	26.4
$\mathcal{L}_{\text{conf}}$ using KL	30.7	35.0	17.7	28.2
w/o Weighted feature terms	29.8	33.9	17.6	28.0
w/o L1 Reg. term	30.5	34.8	17.8	28.6

Table 5. Impact of various finetuning loss components on MS-COCO (10-shot). We finetune DeFRCN without base data using the generator without any regularization techniques.

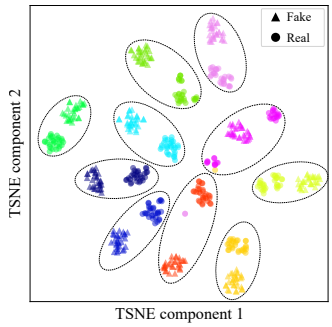
In Tab. 5, we study the impact of various finetuning loss components. In the first row, we start with finetuning DeFRCN with the proposed generator and the novel training loss proposed in the main paper $\mathcal{L}_{\text{N}}$ without any regularization (i.e., CFA, data augmentations, and mEWC). Upon removing the cross-entropy confidence loss $\mathcal{L}_{\text{conf}}$ (row2), we notice that the base performance drop by 2.4 points causing the overall AP and the AR to drop. This denotes that the confidence loss helps generating base base features with higher probabilities at the final softmax layer. If we replace the cross-entropy loss with KL divergence (row3) between the teacher and student logits, we get the same results but with a slight drop in the AR. As for the distillation terms, we show that by removing the weighted feature distillation terms (row 4), the base and overall performance drops. Finally, if the proposed L1 regression distillation term is removed (row 5), we notice a slight decrease in the base and overall performance. Motivated by this ablation, we opt to perform novel training with the overall loss containing the confidence and feature distillation loss terms.

C. Generator Training Analysis

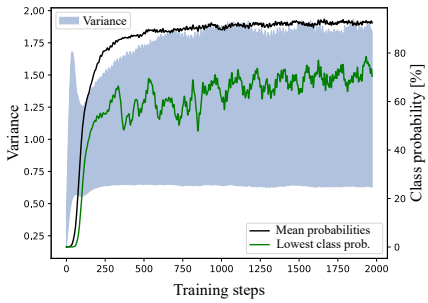
In Fig. 1a, we show a TSNE visualization of the real and fake generated instance-level features for 10 random MS-COCO base classes. We generate 30 features per class to better visualize the generated feature distribution. We show that the forged features are consistently near the real base features (with some overlaps) confirming that the generator is able to depict near-real base feature distribution. Furthermore, we show the features generated using class-agnostic statistics in Fig. 1b. In contrast to the features generated via the class-wise stats, the fake samples are farther away from the real ones.



(a) A TSNE visualization of the generated fake features via class-wise stats and real base features for 10 random classes with 30 features per class to better illustrate the forged features distribution.



(b) A TSNE visualization of the generated fake features via class-agnostic stats and real base features for 10 random classes with 30 features per class.



(c) A highlight of the fake class probabilities along with the features variance. We show the lowest class probability and the mean probabilities across all base classes (MS-COCO).

Figure 1. A TSNE visualization of some generated features via class-wise (a) and class-agnostic (b) stats. The fake class probabilities as well as the features' variance during generator training is illustrated in (c).

Methods / Shots	w/E	w/B	Novel Set 1					Novel Set 2					Novel Set 3				
			1	2	3	5	10	1	2	3	5	10	1	2	3	5	10
FRCN-ft-full [6]	✗	✓	15.2	20.3	29.0	25.5	28.7	13.4	20.6	28.6	32.4	38.8	19.6	20.8	28.7	42.2	42.1
TFA w/ fc [6]	✗	✓	36.8	29.1	43.6	55.7	57.0	18.2	29.0	33.4	35.5	39.0	27.7	33.6	42.5	48.7	50.2
TFA w/ cos [6]	✗	✓	39.8	36.1	44.7	55.7	56.0	23.5	26.9	34.1	35.1	39.1	30.8	34.8	42.8	49.5	49.8
MPSR [8]	✗	✓	42.8	43.6	48.4	55.3	61.2	29.8	28.1	41.6	43.2	47.0	35.9	40.0	43.7	48.9	51.3
DeFRCN [5]	✗	✓	57.0	58.6	64.3	67.8	67.0	35.8	42.7	51.0	54.4	52.9	52.5	56.6	55.8	**60.7**	62.5
Meta R-CNN* [10]	✗	✓	16.8	20.1	20.3	38.2	43.7	7.7	12.0	14.9	21.9	31.1	9.2	13.9	26.2	29.2	36.2
FSRW [3]	✗	✓	14.8	15.5	26.7	33.9	47.2	15.7	15.3	22.7	30.1	39.2	19.2	21.7	25.7	40.6	41.3
MetaDet [7]	✗	✓	18.9	20.6	30.2	36.8	49.6	21.8	23.1	27.8	31.7	43.0	20.6	23.9	29.4	43.9	44.1
FsDetView* [9]	✗	✓	25.4	20.4	37.4	36.1	42.3	22.9	21.7	22.6	25.6	29.2	32.4	19.0	29.8	33.2	39.8
CFA w/ fc [2]	✗	✓	40.0	35.5	40.9	54.1	56.9	22.2	27.1	35.2	38.5	40.9	29.7	35.1	39.5	47.2	51.3
CFA w/ cos [2]	✗	✓	41.2	43.6	49.5	56.5	57.3	21.3	27.4	35.3	39.1	42.1	31.7	39.1	44.6	49.9	52.6
CFA-DeFRCN [2]	✗	✓	**58.2**	**63.3**	**65.8**	**68.9**	**67.1**	**37.1**	**45.5**	**51.3**	**55.2**	**53.8**	**54.7**	**57.8**	**56.9**	60.0	**63.3**
DeFRCN	✗	✗	53.3	47.4	58.7	58.8	59.6	33.0	37.0	49.5	53.8	48.5	47.1	45.8	52.7	52.8	52.6
NIFF-DeFRCN	✗	✗	**62.8**	**67.2**	**68.0**	**70.3**	**68.8**	**38.4**	**42.9**	**54.0**	**56.4**	**54.0**	**56.4**	**62.1**	**61.2**	**64.1**	**63.9**
Retentive R-CNN [1]	✓	✓	42.4	45.8	45.9	53.7	56.1	21.7	27.8	35.2	37.0	40.3	30.2	37.6	43.0	49.7	50.1
CFA w/ fc [2]	✓	✓	39.0	34.9	41.4	54.8	57.0	21.8	26.1	35.3	37.1	40.1	29.9	34.3	40.1	47.0	52.6
CFA w/ cos [2]	✓	✓	42.4	43.9	50.3	56.6	57.3	21.0	27.5	35.3	38.6	41.4	32.3	38.0	44.5	49.8	52.7
CFA-DeFRCN [2]	✓	✓	**59.0**	**63.5**	**66.4**	**68.4**	**68.3**	**37.0**	**45.8**	**50.0**	**54.2**	**52.5**	**54.8**	**58.5**	**56.5**	**61.3**	**63.5**
NIFF-DeFRCN	✓	✗	**63.5**	**67.2**	**68.3**	**71.1**	**69.3**	**37.8**	**41.9**	**53.4**	**56.0**	**53.5**	**55.3**	**60.5**	**61.1**	**63.7**	**63.9**

Table 6. PASCAL-VOC G-FSOD (nAP50) results for $1, 2, 3, 5, 10$-shot settings for all three splits are reported. Similar to [1, 2], w/E denotes the ensemble-based inference paradigm. The best and second-best results are color coded. w/B indicates whether the base data is available during novel finetuning. '-' represents unrecorded results in prior works. '*' represents results reported in [2].

In Fig. 1c, we highlight the quality and diversity of the generated features with respect to the feature variance. We show the mean class probability (black) across all classes for the generated features and the lowest class probability (green). It can be seen that the generator can learn diverse features with a high variance while maintaining high class probabilities with 95% mean class probability and around 75% for the lowest class probability.

D. PASCAL-VOC Novel Results

We report the novel performance on PASCAL-VOC (nAP50) in Tab. 6. We show that adopting NIFF achieves state-of-the-art results with and without the ensemble evaluation protocol. As previously mentioned, although that the performance on novel classes is not our primary objective, NIFF-DeFRCN achieves the state-of-the-art on PASCAL-VOC in all cases but the 2-shot setting in split 2 .

E. Multiple Runs

We run NIFF-DeFRCN using 10 and 30 different seeds for MS-COCO and PASCAL-VOC, respectively. Then, we compare with the baselines [2,5,6]. The aim is to investigate the performance robustness over multiple runs.

MS-COCO. In Tab. 7, we show the results for MS-COCO dataset. We notice that despite the absence of base data, NIFF-DeFRCN consistently achieves a higher AP and bAP over all shot settings.

PASCAL-VOC. In Tab. 8 and Tab. 9, we demonstrate the AP50 and nAP50 results, respectively, for PASCAL-VOC dataset. Similar to MS-COCO, we observe that NIFF-DeFRCN consistently achieves a higher AP50 while delivering competitive results on the nAP50 over various shots.

Methods / Shots	5 shot			10 shot			30 shot		
	AP	bAP	nAP	AP	bAP	nAP	AP	bAP	nAP
TFA w/ fc [6]	25.6±0.5	31.8±0.5	6.9±0.7	26.2±0.5	32.0±0.5	9.1±0.5	28.4±0.3	33.8±0.3	12.0±0.4
TFA w/ cos [6]	25.9±0.6	32.3±0.6	7.0±0.7	26.6±0.5	32.4±0.6	9.1±0.5	28.7±0.4	34.2±0.4	12.1±0.4
CFA w/ fc [2]	29.1±0.3	36.2±0.3	7.7±0.6	29.9±0.3	36.7±0.2	9.6±0.6	30.8±0.2	36.6±0.2	13.6±0.3
CFA w/ cos [2]	29.3±0.2	36.0±0.2	9.2±0.5	30.2±0.2	36.6±0.1	11.2±0.5	31.1±0.1	36.6±0.1	14.8±0.2
DeFRCN [5]	27.8±0.3	32.6±0.3	13.6±0.7	29.7±0.2	34.0±0.2	16.8±0.6	31.4±0.1	34.8±0.1	21.2±0.4
CFA-DeFRCN [2]	28.4±0.2	32.8±0.2	15.2±0.5	30.2±0.2	34.0±0.2	18.8±0.4	31.7±0.1	34.6±0.1	23.0±0.3
NIFF-DeFRCN	31.1±0.1	36.6±0.0	14.6±0.2	32.1±0.1	36.8±0.1	18.0±0.2	33.3±0.0	37.7±0.1	20.0±0.1

Table 7. G-FSOD experimental results for 5,10,30-shot settings on MS-COCO. We report base (bAP), novel (nAP), and overall (AP) for multiple runs using 10 different seeds.

Set	Methods	Shots				
		1	2	3	5	10
All Set 1	CFA w/ fc [2]	66.3±0.8	68.0±0.5	70.1±0.4	71.7±0.5	73.2±0.5
	CFA w/ cos [2]	66.5±0.9	69.2±0.6	71.1±0.6	72.5±0.4	73.4±0.4
	DeFRCN [5]	67.8±1.4	71.3±0.8	72.6±0.5	73.6±0.5	74.1±0.5
	CFA-DeFRCN [2]	69.0±1.4	72.6±0.7	73.1±0.4	74.0±0.5	74.3±0.4
	NIFF-DeFRCN	71.2±0.8	74.2±0.4	75.4±0.4	76.3±0.3	76.7±0.3
All Set 2	CFA w/ fc [2]	64.9±0.9	66.4±0.7	68.3±0.5	69.6±0.3	70.8±0.5
	CFA w/ cos [2]	64.1±0.9	66.5±0.5	68.1±0.5	69.3±0.2	70.4±0.4
	DeFRCN [5]	65.2±1.0	68.0±0.8	69.2±0.6	70.6±0.6	71.3±0.5
	CFA-DeFRCN [2]	66.4±1.0	69.0±0.8	70.4±0.7	71.3±0.7	72.1±0.4
	NIFF-DeFRCN	68.0±0.8	70.5±0.5	71.7±0.5	72.8±0.4	73.7±0.3
All Set 3	CFA w/ fc [2]	65.2±0.8	66.8±0.8	69.1±0.7	70.9±0.6	72.3±0.4
	CFA w/ cos [2]	64.9±1.2	67.5±1.0	69.7±0.8	71.6±0.5	72.7±0.3
	DeFRCN [5]	66.9±2.0	70.6±0.8	71.2±0.6	72.9±0.5	73.5±0.3
	CFA-DeFRCN [2]	68.3±1.6	71.4±0.8	72.3±0.5	73.5±0.5	74.0±0.3
	NIFF-DeFRCN	70.7±0.7	73.7±0.5	74.7±0.4	75.5±0.3	76.3±0.2

Table 8. G-FSOD experimental results for 1,2,3,5,10-shot settings on the three all sets of Pascal VOC (AP50).

Set	Methods	Shots				
		1	2	3	5	10
Novel Set 1	CFA w/ fc [2]	28.2±3.1	35.0±1.9	41.9±1.4	47.8±1.6	53.3±1.6
	CFA w/ cos [2]	30.9±3.9	40.9±2.5	47.8±2.4	53.1±1.4	56.1±1.4
	DeFRCN [5]	43.8±4.3	57.5±2.5	61.4±1.7	65.3±0.9	67.0±1.4
	CFA-DeFRCN [2]	45.4±4.9	60.3±2.2	62.1±1.4	66.4±0.9	67.6±1.2
	NIFF-DeFRCN	46.0±3.0	57.2±1.7	62.0±1.4	65.5±1.1	67.2±1.1
Novel Set 2	CFA w/ fc [2]	20.0±3.5	26.4±2.9	32.8±2.2	37.3±1.7	41.8±1.9
	CFA w/ cos [2]	21.0±3.5	29.0±2.3	34.6±2.3	38.9±1.2	43.0±1.9
	DeFRCN [5]	31.5±3.6	40.9±2.2	45.6±2.0	50.1±1.4	52.9±1.1
	CFA-DeFRCN [2]	32.9±3.7	42.3±2.2	47.1±1.9	51.2±1.4	55.3±1.3
	NIFF-DeFRCN	30.1±3.0	39.6±1.8	45.0±1.9	49.4±1.6	52.8±1.3
Novel Set 3	CFA w/ fc [2]	20.3±3.4	26.4±3.1	34.3±2.5	41.2±2.4	46.5±1.6
	CFA w/ cos [2]	21.5±4.7	30.4±4.1	38.4±2.8	45.5±2.1	49.9±1.0
	DeFRCN [5]	38.2±6.8	50.9±2.8	54.1±2.2	59.2±1.2	61.9±1.3
	CFA-DeFRCN [2]	41.4±5.8	52.9±3.0	56.1±1.7	60.3±1.1	62.9±0.9
	NIFF-DeFRCN	41.1±2.6	52.5±1.8	56.4±1.5	59.7±1.2	62.1±1.0

Table 9. G-FSOD experimental results for 1,2,3,5,10-shot settings on the three novel sets of Pascal VOC (nAP50).

F. Qualitative Results

We present various qualitative results in Fig. 2 on the MS-COCO (10-shot). In the first column, we show images with only base classes (green boxes) followed by images with only novel classes (blue boxes) in the second column. In the third column, we present images with both classes. We chose to present these three cases to validate the performance of the proposed NIFF in various scenarios. . Further, we also present various failure cases in the last two columns.

Figure 2. Qualitative results of the proposed NIFF method (NIFF-DeFRCN) on the MS-COCO(10-shot) dataset. Success scenarios are demonstrated in the first three columns show while the last two columns present the failure scenarios. Base classes are denoted by green bounding boxes while novel classes are colored with blue.

References

[1] Zhibo Fan, Yuchen Ma, Zeming Li, and Jian Sun. Generalized few-shot object detection without forgetting. In IEEE Conference on Computer Vision and Pattern Recognition, pages 4527–4536, 2021. 3

[2] Karim Guirguis, Ahmed Hendawy, George Eskandar, Mohamed Abdelsamad, Matthias Kayser, and Jürgen Beyerer. Cfa: Constraint-based finetuning approach for generalized few-shot object detection. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 4038–4048, 2022. 1, 3, 4

[3] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. In IEEE International Conference on Computer Vision, pages 8419–8428, 2018. 3

[4] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences, 114, 2016. 2

[5] Limeng Qiao, Yuxuan Zhao, Zhiyuan Li, Xi Qiu, Jianan Wu, and Chi Zhang. DeFRCN: Decoupled faster R-CNN for few-shot object detection. In IEEE International Conference on Computer Vision, 2021. 1, 3, 4

[6] Xin Wang, Thomas E. Huang, Trevor Darrell, Joseph E. Gonzalez, and Fisher Yu. Frustratingly simple few-shot object detection. In International Conference on Machine Learning, pages 9919–9928, 2020. 3, 4

[7] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Meta-learning to detect rare objects. In IEEE International Conference on Computer Vision, pages 9924–9933, 2019. 3

[8] Jiaxi Wu, Songtao Liu, Di Huang, and Yunhong Wang. Multi-scale positive sample refinement for few-shot object detection. In European Conference on Computer Vision, pages 456–472, 2020. 3

[9] Yang Xiao and Renaud Marlet. Few-shot object detection and viewpoint estimation for objects in the wild. In European Conference on Computer Vision, pages 192–210, 2020. 3

[10] Xiaopeng Yan, Ziliang Chen, Anni Xu, Xiaoxi Wang, Xiaodan Liang, and Liang Lin. Meta R-CNN: Towards general solver for instance-level low-shot learning. In IEEE International Conference on Computer Vision, pages 9577–9586, 2019. 3